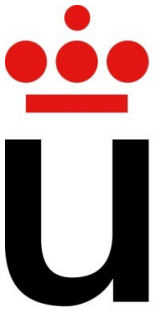


Universidad Rey Juan Carlos

ESCUELA TÉCNICA SUPERIOR
DE INGENIERÍA INFORMÁTICA

Asignatura: APRENDIZAJE AUTOMÁTICO I
Grado en Ciencia e Ingeniería de Datos

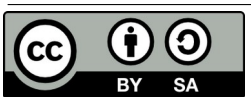
Ejercicios de la asignatura

(Fecha del material: Diciembre 2023)

Curso académico 2023-2024

Material docente en abierto de la Universidad Rey Juan Carlos

Autores: Isaac Martín de Diego, Carmen Lancho



Índice de los ejercicios

Además del presente documento, se ponen a disposición del lector los códigos de R que generan tanto los enunciados como las soluciones con código R en el siguiente repositorio:

https://github.com/URJCDSLab/EjerciciosAprendizajeAutomatico

Aprendizaje Automático I
Grado en Ciencia e Ingeniería de Datos
Universidad Rey Juan Carlos

Ejercicio: Proyectos de Ciencia de Datos

DSLab

diciembre, 2023

Este ejercicio consiste en localizar un proyecto de Ciencia de Datos en alguna de las siguientes referencias:

- Top 10 Data Science Case Studies Projects with Examples and Solutions in Python to inspire your data science learning in 2023.
- Data Science Case Studies: Solved and Explained
- Data in Action: 7 Data Science Case Studies Worth Reading
- Top 8 Data Science Case Studies for Data Science Enthusiasts

Contesta a las siguientes preguntas:

1. ¿Cuál es el problema que se trata de resolver? ¿Qué preguntas de negocio son las más importantes?
2. ¿Cuál es el origen de los datos? Existían antes de plantear la pregunta o fueron resultado de una técnica de muestreo.
3. ¿Qué has podido averiguar sobre el proceso de preparación y enriquecimiento de los datos?
4. ¿Qué técnicas de Aprendizaje Automático (Supervisado o no supervisado) han sido empleadas?
5. ¿Cuál ha sido el rendimiento del modelo?

6. ¿Has podido averiguar cómo se presentaron los datos al cliente?
7. Finalmente, ¿se puso en producción el modelo resultante? ¿sigue funcionando?

Aprendizaje Automático I
Grado en Ciencia e Ingeniería de Datos
Universidad Rey Juan Carlos

Solución ejercicio: Proyectos de Ciencia de Datos

DSLab

diciembre, 2023

Por ejemplo, consideremos el siguiente caso de éxito:

Detección del cáncer de mama metastásico

1. ¿Cuál es el problema que se trata de resolver? ¿Qué preguntas de negocio son las más importantes?

Las metástasis de un tumor influyen en las decisiones terapéuticas de diversos tipos de cáncer. La identificación histológica de las células tumorales en los ganglios linfáticos puede ser laboriosa y propensa a errores, especialmente en el caso de focos tumorales pequeños.

El objetivo es detectar cáncer de mama metastásico en biopsias de ganglio linfático centinela.

2. ¿Cuál es el origen de los datos? Existían antes de plantear la pregunta o fueron resultado de una técnica de muestreo.

Para lleva a cabo el estudio, se obtuvieron imágenes de ganglios linfáticos teñidos con hematoxilina-eosina de 399 pacientes (conjunto de datos de desafío Camelyon16 que dominio público).

3. ¿Qué has podido averiguar sobre el proceso de obtención y preparación de los datos?

El algoritmo se desarrolló utilizando 270 muestras de entrenamiento y se evaluó en los 129 pacientes restantes. Además, se compararon los resultados con los obtenidos en un laboratorio independiente.

4. ¿Qué técnicas de Aprendizaje Automático (Supervisado o no supervisado) han sido empleadas?

El algoritmo empleado es una red neuronal profunda, Deep Learning, denominada LYNA: "LYmph Node Assistant". Adjuntamos archivo con la descripción del algoritmo.

5. ¿Cuál ha sido el rendimiento del modelo?

El modelo propuesto, LYNA presenta un "área bajo la curva" de 99% y una sensibilidad del 91%.

6. ¿Has podido averiguar cómo se presentaron los datos al cliente?

Se presentan la metodología, el algoritmo y sus resultados en la revista científica *Archives of Pathology & Laboratory Medicne*.

7. Finalmente, ¿se puso en producción el modelo resultante? ¿sigue funcionando?

Este suele ser un dato del que no se dispone, sin embargo parece ser que el modelo propuesto puede ser utilizado por diversas instituciones.

Referencias:

- https://meridian.allenpress.com/aplm/article/143/7/859/10038/Artificial-Intelligence-Based-Breast-Cancer-Nodal
- https://allen.silverchair-cdn.com/allen/content_public/journal/aplm/143/7/10.5858_arpa.2018-0147-oa/2/arpa_2018-0147-oa.pdf?Expires=1701888792&Signature=kYuABkfH35mSKkDqED9zmS32Tqum6TQV8E4aEJGU6lS1LLjx9oujKxgVFD4BS0ja6zV~mV3VVrKXQxG~jYSZPthRozu73hmaXTtxAk7C6TajthTCqB1mo~XTS1A5J~rM7u6OAObs3Upvhr4cwhyLIkcCHrhMhBWVQgVaKM3CESTumJzkAx6X3IvHh8hWR3LsTiOl1nswcCJ9ntg~LRsRBNq8tt5LQJSoR-bKVX2ASBhcU-vVpLtA9bjqQSc8oG7hDUtoby8oUsMN5XBKrXGqUf5Ss1iZwyxQ-YfbThJcDkeD9LKZtN1DV0TF-DIlXBowy4ib6ZwHh9OO34fG3yLC8Q__&Key-Pair-Id=APKAIE5G5CRDK6RD3PGA

DSLAB
DATA SCIENCE LABORATORY

Aprendizaje Automático I
Grado en Ciencia e Ingeniería de Datos
Universidad Rey Juan Carlos

Ejercicio: Particiones sobre los datos

DSLab

diciembre, 2023

En este ejercicio vamos a trabajar sobre las particiones de un conjunto de datos.

1. En primer lugar, debes leer el conjunto de datos `adult` de la página web de UC Irvine Machine Learning Repository.
2. ¿Qué tamaño tiene la base de datos que has leído?
3. Implementa tu propio código en R para dividir la base de datos en 3 muestras de:
   - Entrenamiento 60%
   - Prueba 20%
   - Validación 20%
4. ¿Qué número de observaciones tienen cada una de las particiones?
5. Calcula la media de la variable `Area` en la muestra de entrenamiento. ¿Coincide con el valor en la muestra de validación?
6. Convierte la variable `Area` en la muestra de entrenamiento en otra variable de media 0 y varianza 1. Aplica la misma transformación en las otras particiones, pero cuidado, empleando la media y la varianza de la partición de entrenamiento. Recuerda que, cuando tienes un nuevo dato, no dispones de estadísticos en "todos" los nuevos datos. ¿Qué valores obtienes para la media y varianza de las nuevas variables en las particiones de validación y prueba?

December 15, 2023

Solución ejercicio: Particiones sobre los datos

Los siguientes resultados han sido obtenidos con un script de R.

```r
# Librería para leer XLSX
library(readxl)
library(dplyr)

# leemos los datos previamente salvados
df <- read_excel("Dry_Bean_Dataset.xlsx")

# Número de observaciones y variables
dim(df)
```

```
## [1] 13611    17
```

```r
# Particiones

# mediante una semilla conseguimos que el ejercicio sea reproducible
set.seed(12321)

# creamos índices
ntotal <- dim(df)[1]
indices <- 1:ntotal
ntrain <- ntotal * .6
ntest <- ntotal *.2
indices.train <- sample(indices, ntrain, replace = FALSE)
indices.test  <- sample(indices[-indices.train],ntest,replace=FALSE)
indices.valid <- indices[-c(indices.train,indices.test)]

# Usamos el 60% de la base de datos como conjunto de entrenamiento, 20% como conjunto de test y 20% como
train  <- df[indices.train, ]
test   <- df[indices.test, ]
valid  <- df[indices.valid,]

dim(train)
```

```
## [1] 8166   17
```

```r
dim(test)
```

```
## [1] 2722   17
```

```r
dim(valid)
```

```
## [1] 2723   17
```

```r
# Media de la variable AREA
media.train=mean(train$Area)
sd.train=sqrt(var(train$Area))
mean(test$Area)

## [1] 52377.31

mean(valid$Area)

## [1] 53244.92

# Escalamos la variable.

train=
  train %>%
  mutate(Area_Scale=scale(Area))

mean(train$Area_Scale)

## [1] 1.944833e-17

var(train$Area_Scale)

##      [,1]
## [1,]    1

# Aplicamos la misma transformación en los datos de test y validación

test=
  test %>%
  mutate(Area_Scale=(Area-media.train)/sd.train)

mean(test$Area_Scale)

## [1] -0.02785735

var(test$Area_Scale)

## [1] 0.8295407

valid=
  valid %>%
  mutate(Area_Scale=(Area-media.train)/sd.train)

mean(valid$Area_Scale)

## [1] 0.001295361

var(valid$Area_Scale)

## [1] 1.024713
```

Información de la sesión de R (incluyendo información sobre el sistema operativo, la versión de R y los paquetes usados):

```r
sessionInfo()
```

```
## R version 4.3.1 (2023-06-16)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 20.04.6 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/atlas/libblas.so.3.10.3
## LAPACK: /usr/lib/x86_64-linux-gnu/atlas/liblapack.so.3.10.3;  LAPACK version 3.9.0
##
## locale:
##  [1] LC_CTYPE=es_ES.UTF-8       LC_NUMERIC=C               LC_TIME=es_ES.UTF-8
##  [4] LC_COLLATE=es_ES.UTF-8     LC_MONETARY=es_ES.UTF-8    LC_MESSAGES=es_ES.UTF-8
##  [7] LC_PAPER=es_ES.UTF-8       LC_NAME=C                  LC_ADDRESS=C
## [10] LC_TELEPHONE=C             LC_MEASUREMENT=es_ES.UTF-8 LC_IDENTIFICATION=C
##
## time zone: Europe/Madrid
## tzcode source: system (glibc)
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] readxl_1.4.3     caret_6.0-94     lattice_0.21-9   ggplot2_3.4.3    rpart.plot_3.1.1
## [6] rpart_4.1.19     caTools_1.18.2   dplyr_1.1.3      ISLR2_1.3-2
##
## loaded via a namespace (and not attached):
##  [1] gtable_0.3.4         xfun_0.40            recipes_1.0.8        tzdb_0.4.0
##  [5] vctrs_0.6.3          tools_4.3.1          bitops_1.0-7         generics_0.1.3
##  [9] stats4_4.3.1         parallel_4.3.1       proxy_0.4-27         tibble_3.2.1
## [13] fansi_1.0.5          highr_0.10           ModelMetrics_1.2.2.2 pkgconfig_2.0.3
## [17] Matrix_1.6-1.1       data.table_1.14.8    lifecycle_1.0.3      stringr_1.5.0
## [21] compiler_4.3.1       farver_2.1.1         tinytex_0.47         munsell_0.5.0
## [25] codetools_0.2-19     htmltools_0.5.6.1    class_7.3-22         yaml_2.3.7
## [29] prodlim_2023.08.28   pillar_1.9.0         MASS_7.3-60          gower_1.0.1
## [33] iterators_1.0.14     foreach_1.5.2        nlme_3.1-163         parallelly_1.36.0
## [37] lava_1.7.2.1         tidyselect_1.2.0     digest_0.6.33        stringi_1.7.12
## [41] future_1.33.0        reshape2_1.4.4       purrr_1.0.2          listenv_0.9.0
## [45] labeling_0.4.3       splines_4.3.1        fastmap_1.1.1        grid_4.3.1
## [49] colorspace_2.1-0     cli_3.6.1            magrittr_2.0.3       survival_3.5-7
## [53] utf8_1.2.3           e1071_1.7-13         future.apply_1.11.0  readr_2.1.4
## [57] withr_2.5.1          scales_1.2.1         lubridate_1.9.3      timechange_0.2.0
## [61] rmarkdown_2.25       globals_0.16.2       nnet_7.3-19          timeDate_4022.108
## [65] cellranger_1.1.0     hms_1.1.3            evaluate_0.22        knitr_1.44
## [69] hardhat_1.3.0        rlang_1.1.1          Rcpp_1.0.11          glue_1.6.2
## [73] pROC_1.18.4          ipred_0.9-14         rstudioapi_0.15.0    R6_2.5.1
## [77] plyr_1.8.9
```

```r
Sys.time()
```

```
## [1] "2023-11-01 20:39:27 CET"
```

Aprendizaje Automático I
Grado en Ciencia e Ingeniería de Datos
Universidad Rey Juan Carlos

Ejercicio: EDA

DSLab

diciembre, 2023

Utilizar el dataframe denominado *airquality* de la librería de R *datasets* para responder a las siguientes cuestiones:

- ¿Cuántos campos y observaciones tiene el dataframe?. Utilizar las funciones *head* y *dim*.
- Evaluar el dataframe con la función *summary*
  - ¿Tiene observaciones con elementos faltantes (NA)?
  - ¿A qué meses corresponden las observaciones?
- Temperatura máxima del viento en el mes de **mayo**.
- Media del ozono en el mes de Julio (*Ojo con los NA*).
- Mes donde la temperatura fue mayor.
- Número de observaciones donde la temperatura fue mayor que 90 y además el ozono fue menos que 100 filtrando por mes (utilizar la función *length*).
- Haciendo un estudio de los datos. ¿Qué podemos concluir? ¿Existe alguna relación entre las variables *Ozono*,*Temperatura*, y *Radiación Solar*? Se recomienda hacer la media mes a mes de cada variable.

Solución ejercicio: EDA

Los siguientes resultados han sido obtenidos con un script de R.

```
# • Cuantos campos y observaciones tiene el dataframe.  Utilizar "head" y "dim".
head(airquality) # -> Hay 6 campos: Ozone Solar.R Wind Temp Month Day.

##      Ozone  Solar.R Wind Temp Month Day
## 1 41.00000 190.0000  7.4   67     5   1
## 2 36.00000 118.0000  8.0   72     5   2
## 3 12.00000 149.0000 12.6   74     5   3
## 4 18.00000 313.0000 11.5   62     5   4
## 5 59.11538 181.2963 14.3   56     5   5
## 6 28.00000 181.2963 14.9   66     5   6

dim(airquality)  # -> 153 observaciones con 6 campos.

## [1] 153   6

# • Evaluar el dataframe con la instrucción "summary".
#     o ¿Tiene observaciones con elementos nulos (NA)?
#     o ¿A que meses corresponden las observaciones?
summary(airquality)

##      Ozone           Solar.R           Wind             Temp           Month
##  Min.   :  1.00   Min.   :  7.0   Min.   : 1.700   Min.   :56.00   Min.   :5.000
##  1st Qu.: 21.00   1st Qu.:118.5   1st Qu.: 7.400   1st Qu.:72.00   1st Qu.:6.000
##  Median : 45.00   Median :199.0   Median : 9.700   Median :79.00   Median :7.000
##  Mean   : 46.24   Mean   :185.8   Mean   : 9.958   Mean   :77.88   Mean   :6.993
##  3rd Qu.: 59.12   3rd Qu.:257.5   3rd Qu.:11.500   3rd Qu.:85.00   3rd Qu.:8.000
##  Max.   :168.00   Max.   :334.0   Max.   :20.700   Max.   :97.00   Max.   :9.000
##                   NA's   :3
##       Day
##  Min.   : 1.0
##  1st Qu.: 8.0
##  Median :16.0
##  Mean   :15.8
##  3rd Qu.:23.0
##  Max.   :31.0
##

# Hay valores NA en Ozone (37) y en Solar Radiation (7).
# Los meses durante los que se realizaron las observaciones son del 5 al 9 (es decir de mayo a septiembr

# • Temperatura máxima del viento en el mes de mayo.
max(airquality[airquality$Month == 5,]$Temp) # <- 81

## [1] 81

# • Media del ozono en el mes de Julio.

mean(airquality[airquality$Month == 7,]$Ozone,na.rm = TRUE) # -> 59.11538

## [1] 59.11538
```

```r
media=mean(airquality[airquality$Month == 7,]$Ozone,na.rm = TRUE) # -> 59.11538

# Transformar al valor de la media los NA.
airquality$Ozone[is.na(airquality$Ozone)] <- media
# Estudiar el efecto de esta asignación sobre la desviación típica

mean(airquality[airquality$Month == 7,]$Ozone)
```

```
## [1] 59.11538
```

```r
# • Mes donde la temperatura fue mayor.
airquality[max(airquality$Temp),]$Month # -> Agosto (8)
```

```
## [1] 8
```

```r
# • Mes donde la temperatura y el ozono fue mayor.
length(airquality[airquality$Temp > 90 & airquality$Ozone < 100,"Month"]) # -> 13
```

```
## [1] 13
```

```r
# • Haciendo un estudio de los datos, ¿Qué podemos concluir?
# ¿Existe alguna relación entre las variables Ozono, Temperatura y Radiación Solar?
# Se recomienda hacer la media mes a mes de cada variable.

mean(airquality$Ozone[airquality$Month == 5])
```

```
## [1] 29.34119
```

```r
mean(airquality$Ozone[airquality$Month == 6])
```

```
## [1] 50.2141
```

```r
mean(airquality$Ozone[airquality$Month == 7])
```

```
## [1] 59.11538
```

```r
mean(airquality$Ozone[airquality$Month == 8])
```

```
## [1] 59.82506
```

```r
mean(airquality$Ozone[airquality$Month == 9])
```

```
## [1] 32.37051
```

```r
mean(airquality$Temp[airquality$Month == 5])
```

```
## [1] 65.54839
```

```r
mean(airquality$Temp[airquality$Month == 6])
```

```
## [1] 79.1
```

```r
mean(airquality$Temp[airquality$Month == 7])
```

```
## [1] 83.90323
```

```r
mean(airquality$Temp[airquality$Month == 8])
```

```
## [1] 83.96774
```

```r
mean(airquality$Temp[airquality$Month == 9])
```

```
## [1] 76.9
```

```r
media_Solar.R_5=mean(airquality$Solar.R[airquality$Month == 5],na.rm = TRUE)
sqrt(var(airquality$Solar.R[airquality$Month == 5],na.rm=TRUE))
```

```
## [1] 107.1295
```

```r
media_Solar.R_6=mean(airquality$Solar.R[airquality$Month == 6],na.rm = TRUE)
media_Solar.R_7=mean(airquality$Solar.R[airquality$Month == 7],na.rm = TRUE)
media_Solar.R_8=mean(airquality$Solar.R[airquality$Month == 8],na.rm = TRUE)
media_Solar.R_9=mean(airquality$Solar.R[airquality$Month == 9],na.rm = TRUE)

# Transformar los NA.
table(is.na(airquality$Solar.R))
```

```
##
## FALSE  TRUE
##   150     3
```

```r
indices5=which(is.na(airquality$Solar.R[airquality$Month == 5]))
# ojo, estamos cambiando todos los datos sin haber salvado la anterior versión
# del dataframe
airquality$Solar.R[airquality$Month == 5][indices5]=media_Solar.R_5
mean(airquality$Solar.R[airquality$Month == 5])
```

```
## [1] 181.2963
```

```r
sqrt(var(airquality$Solar.R[airquality$Month == 5]))
```

```
## [1] 107.1295
```

```r
# Si se hace la media del ozono, temperatura y radiacion solar podemos observar como más o menos todos
# El mes de junio es el único que presenta algo de variación.
# Se podría concluir que todas las variables indicadas tiene relación entre ellas.
```

Información de la sesión de R (incluyendo información sobre el sistema operativo, la versión de R y los paquetes usados):

```r
sessionInfo()
```

```
## R version 4.3.1 (2023-06-16)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 20.04.6 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/atlas/libblas.so.3.10.3
## LAPACK: /usr/lib/x86_64-linux-gnu/atlas/liblapack.so.3.10.3;  LAPACK version 3.9.0
##
## locale:
##  [1] LC_CTYPE=es_ES.UTF-8       LC_NUMERIC=C               LC_TIME=es_ES.UTF-8
##  [4] LC_COLLATE=es_ES.UTF-8     LC_MONETARY=es_ES.UTF-8    LC_MESSAGES=es_ES.UTF-8
##  [7] LC_PAPER=es_ES.UTF-8       LC_NAME=C                  LC_ADDRESS=C
## [10] LC_TELEPHONE=C             LC_MEASUREMENT=es_ES.UTF-8 LC_IDENTIFICATION=C
##
```

```
## time zone: Europe/Madrid
## tzcode source: system (glibc)
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] knitr_1.44       factoextra_1.0.7 ggplot2_3.4.3    arules_1.7-6     Matrix_1.6-1.1
##
## loaded via a namespace (and not attached):
##  [1] gtable_0.3.4         xfun_0.40            recipes_1.0.8        ggrepel_0.9.3
##  [5] lattice_0.21-9       vctrs_0.6.3          tools_4.3.1          generics_0.1.3
##  [9] stats4_4.3.1         parallel_4.3.1       tibble_3.2.1         fansi_1.0.5
## [13] highr_0.10           pkgconfig_2.0.3      ModelMetrics_1.2.2.2 data.table_1.14.8
## [17] lifecycle_1.0.3      farver_2.1.1         compiler_4.3.1       stringr_1.5.0
## [21] munsell_0.5.0        codetools_0.2-19     DALEX_2.4.3          htmltools_0.5.6.1
## [25] class_7.3-22         yaml_2.3.7           prodlim_2023.08.28   pillar_1.9.0
## [29] MASS_7.3-60          gower_1.0.1          iterators_1.0.14     rpart_4.1.19
## [33] foreach_1.5.2        nlme_3.1-163         parallelly_1.36.0    lava_1.7.2.1
## [37] tidyselect_1.2.0     digest_0.6.33        stringi_1.7.12       future_1.33.0
## [41] dplyr_1.1.3          reshape2_1.4.4       purrr_1.0.2          listenv_0.9.0
## [45] labeling_0.4.3       splines_4.3.1        cowplot_1.1.1        fastmap_1.1.1
## [49] grid_4.3.1           colorspace_2.1-0     cli_3.6.1            magrittr_2.0.3
## [53] survival_3.5-7       utf8_1.2.3           future.apply_1.11.0  withr_2.5.1
## [57] scales_1.2.1         xgboost_1.7.5.1      lubridate_1.9.3      timechange_0.2.0
## [61] rmarkdown_2.25       globals_0.16.2       nnet_7.3-19          timeDate_4022.108
## [65] evaluate_0.22        hardhat_1.3.0        caret_6.0-94         rlang_1.1.1
## [69] Rcpp_1.0.11          glue_1.6.2           pROC_1.18.4          ipred_0.9-14
## [73] rstudioapi_0.15.0    jsonlite_1.8.7       R6_2.5.1             plyr_1.8.9
```

```
Sys.time()
```

```
## [1] "2023-10-31 22:29:26 CET"
```

Aprendizaje Automático I
Grado en Ciencia e Ingeniería de Datos
Universidad Rey Juan Carlos

Ejercicio: EDA2

DSLab

diciembre, 2023

Ejercicio 1

Crear el siguiente *dataframe* mediante estas instrucciones

```
set.seed(1234)

stocks = data.frame(time = as.Date("2009-01-01") + 0:9,
                    Wallmart = rnorm(10,20,1),
                    Target = rnorm(10,20,2),
                    Walgreens = rnorm (10,20,4)
                    )
stocks
```

```
##          time Wallmart   Target Walgreens
## 1  2009-01-01 18.79293 19.04561  20.53635
## 2  2009-01-02 20.27743 18.00323  18.03726
## 3  2009-01-03 21.08444 18.44749  18.23781
## 4  2009-01-04 17.65430 20.12892  21.83836
## 5  2009-01-05 20.42912 21.91899  17.22512
## 6  2009-01-06 20.50606 19.77943  14.20718
```

```
## 7  2009-01-07 19.42526 18.97798  22.29902
## 8  2009-01-08 19.45337 18.17761  15.90538
## 9  2009-01-09 19.43555 18.32566  19.93945
## 10 2009-01-10 19.10996 24.83167  16.25621
```

A continuación, realizar las siguientes operaciones de limpieza de datos:

- Como se puede observar, hay un problema de clave-valor en las compañías con sus observaciones. Por lo tanto, se pide transformar los datos para que tengan una clave "*stock*" y un valor "*precio*". Utilizar la instrucción "*gather*".
- Devolver el dataframe al estado original empleando la instrucción *spread*.
- Utilizando el operador tubería %>% se desea realizar las siguientes operaciones anidadas:
  - Transformar los datos para que tengan una clave "*stock*" y el calor sea el "*precio*". Utilizar la instrucción "*gather*".
  - Agrupar los datos por la clave "*stock*" mediante la instrucción "*group_by*".
  - Obtener el precio mínimo y el máximo utilizando la instrucción "*summarise*".

Ejercicio 2

En este ejercicio vamos a manejar datos contenidos en distintos *dataframes* y operar sobre ellos con *dplyr*.

1. Descargar el paquete *nycflights13*.
2. Evaluar el contenido de los dataframes proporcionados por el paquete. Utilizar *head* y *summary*.
3. Simplificar los dataframes originales a 100 observaciones mediante el comando *head*. Asignarlos a una variable que indique el tipo de dataframe añadiendo la coletilla "*_simple*". *Ejemplo:"*flights_simple*".
4. Selecciona los tipos de aerolínea ("*carrier*") mediante la instrucción *select* y el operador *unique* concatenados con el operador tubería %>%. (Utilizar "*airlines_simple*").
5. Obtener la media y el número máximo de asientos ("*seats*") que tienen los aviones. Utilizar el operador tubería %>% y la instrucción *summarise*.
6. Ordenar los aviones por su número de motores ("*engines*") y número de asientos ("*seats*"). Utilizar la instrucción *arrange*.
7. Averigua qué número de cola ("*tailnum*") comparten los dataframes "*flights_simple*" y "*planes_simple*" que has creado anteriormente. Obten su aerolínea ("*carrier*"). Utilizar la instrucción *inner_join*.
8. Cruzar los datos de vuelos ("*flights*") con los aviones ("*planes*") por el número de cola ("*tailnum*") que no coincidan (usar la instrucción *anti_join*). De esos obtener aquellos con 2 o más motores(usar la instrucción *filter*). Finalmente obtener los dis-

tintos modelos de avión que satisfacen las premisas anteriores (usar la instrucción *unique*).

9. Crea una nueva variable (“*total_delay*”) que calcule el retraso total sumando los delays acumulados (“*dep_delay*”) y (“*arr_delay*”). Utilizar la instrucción *mutate*. Almacena el dataframe resultante en “*flights_total*”.
10. En base a la variable anteriormente obtenida (“*total_delay*”), devuelve los aviones que han llegado con antelación a su destino, es decir aquellos tal que la variable *total_delay* tiene valores negativos.

December 15, 2023

Solución ejercicio: EDA2

Los siguientes resultados han sido obtenidos con un script de R.

```r
# Ejercicio 1 (tidyr y dplyr)
library(tidyr)

##
## Attaching package: 'tidyr'
## The following objects are masked from 'package:Matrix':
##
##     expand, pack, unpack

# A partir del siguiente dataframe realizar las siguientes operaciones de limpieza de datos:
set.seed(1)
stocks <- data.frame(
  time = as.Date('2009-01-01') + 0:9,
  Walmart = rnorm(10, 20, 1),
  Target = rnorm(10, 20, 2),
  Walgreens = rnorm(10, 20, 4)
)
#        time  Walmart   Target Walgreens
# 1  2009-01-01 19.37355 23.02356  23.67591
# 2  2009-01-02 20.18364 20.77969  23.12855
# 3  2009-01-03 19.16437 18.75752  20.29826
# 4  2009-01-04 21.59528 15.57060  12.04259
# 5  2009-01-05 20.32951 22.24986  22.47930
# 6  2009-01-06 19.17953 19.91013  19.77549
# 7  2009-01-07 20.48743 19.96762  19.37682
# 8  2009-01-08 20.73832 21.88767  14.11699
# 9  2009-01-09 20.57578 21.64244  18.08740
# 10 2009-01-10 19.69461 21.18780  21.67177

# Como se puede observar hay un problema de clave-valor en las compañias con sus observaciones.
# Transformar los datos para que tengan una clave stock y el valor sea el precio.
# Por lo tanto se requiere la funcion "gather".

# Opcion 1:
new_stocks <- gather(data = stocks, key = stock, value = price, Walmart, Target, Walgreens)

# Opcion 2:
new_stocks <- gather(data = stocks, key = stock, value = price, Walmart:Walgreens)

# Opcion 3:
new_stocks <- gather(data = stocks, key = stock, value = price, -time)
# El último argumento, -time, significa que todas las columnas excepto el tiempo contienen los pares cl
```

```r
# Devolver el dataframe al estado original utilizando la funcion "spread".
original_stocks <- spread(data = new_stocks, key = stock, value = price)

# Utilizando el operador tuberia %>% se desea realizar las siguientes operaciones anidadas.
# 1) Transformar los datos para que tengan una clave stock y el valor sea el precio mediante la funcion
# 2) Agrupar los datos por la clave stock mediante la funcion "group_by".
# 3) Obtener el precio minimo y maximo utilizando la funcion "summarise".

stocks %>%
  gather(key = stock, value = price,Walmart:Walgreens)%>%
  group_by(stock) %>%
  summarise(min = min(price), max = max(price))
```

```
## Error in summarise(., min = min(price), max = max(price)):  no se pudo encontrar la función
"summarise"
```

```r
########################################################################

# Ejercicio 2 (dplyr)

library(dplyr)
```

```
##
## Attaching package:  'dplyr'
## The following objects are masked from 'package:arules':
##
##     intersect, recode, setdiff, setequal, union
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(nycflights13)

# COMPROBACION.
# Observamos los distintos dataframes que nos proporcionan.
# Utilizamos el nombre del paquete y doblemente dos puntos (::) para comprobarlo.
# Tambien se puede utilizar el nombre del dataframe si previamente estamos familiarizados.

# PRIMERA OBSERVACION.
# Comprobamos las variables de cada uno de los datasets que nos proporcionan mediante la instrucción "he
print(head(flights))
```

```
## # A tibble: 6 x 19
##    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay
##   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl>
## 1  2013     1     1      517            515         2      830            819        11
## 2  2013     1     1      533            529         4      850            830        20
## 3  2013     1     1      542            540         2      923            850        33
## 4  2013     1     1      544            545        -1     1004           1022       -18
## 5  2013     1     1      554            600        -6      812            837       -25
## 6  2013     1     1      554            558        -4      740            728        12
```

```
## # i 10 more variables: carrier <chr>, flight <int>, tailnum <chr>, origin <chr>,
## #   dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>,
## #   time_hour <dttm>

print(head(airports))

## # A tibble: 6 x 8
##   faa   name                            lat   lon   alt    tz dst   tzone
##   <chr> <chr>                         <dbl> <dbl> <dbl> <dbl> <chr> <chr>
## 1 04G   Lansdowne Airport              41.1 -80.6  1044    -5 A     America/New_York
## 2 06A   Moton Field Municipal Airport  32.5 -85.7   264    -6 A     America/Chicago
## 3 06C   Schaumburg Regional            42.0 -88.1   801    -6 A     America/Chicago
## 4 06N   Randall Airport                41.4 -74.4   523    -5 A     America/New_York
## 5 09J   Jekyll Island Airport          31.1 -81.4    11    -5 A     America/New_York
## 6 0A9   Elizabethton Municipal Airport 36.4 -82.2  1593    -5 A     America/New_York

print(head(weather))

## # A tibble: 6 x 15
##   origin  year month   day  hour  temp  dewp humid wind_dir wind_speed wind_gust precip
##   <chr>  <int> <int> <int> <int> <dbl> <dbl> <dbl>    <dbl>      <dbl>     <dbl>  <dbl>
## 1 EWR     2013     1     1     1  39.0  26.1  59.4      270      10.4         NA      0
## 2 EWR     2013     1     1     2  39.0  27.0  61.6      250       8.06        NA      0
## 3 EWR     2013     1     1     3  39.0  28.0  64.4      240      11.5         NA      0
## 4 EWR     2013     1     1     4  39.9  28.0  62.2      250      12.7         NA      0
## 5 EWR     2013     1     1     5  39.0  28.0  64.4      260      12.7         NA      0
## 6 EWR     2013     1     1     6  37.9  28.0  67.2      240      11.5         NA      0
## # i 3 more variables: pressure <dbl>, visib <dbl>, time_hour <dttm>

print(head(airlines))

## # A tibble: 6 x 2
##   carrier name
##   <chr>   <chr>
## 1 9E      Endeavor Air Inc.
## 2 AA      American Airlines Inc.
## 3 AS      Alaska Airlines Inc.
## 4 B6      JetBlue Airways
## 5 DL      Delta Air Lines Inc.
## 6 EV      ExpressJet Airlines Inc.

print(head(planes))

## # A tibble: 6 x 9
##   tailnum  year type                    manufacturer     model  engines seats speed engine
##   <chr>   <int> <chr>                   <chr>            <chr>    <int> <int> <int> <chr>
## 1 N10156   2004 Fixed wing multi engine EMBRAER          EMB-1~       2    55    NA Turbo~
## 2 N102UW   1998 Fixed wing multi engine AIRBUS INDUSTRIE A320-~       2   182    NA Turbo~
## 3 N103US   1999 Fixed wing multi engine AIRBUS INDUSTRIE A320-~       2   182    NA Turbo~
## 4 N104UW   1999 Fixed wing multi engine AIRBUS INDUSTRIE A320-~       2   182    NA Turbo~
## 5 N10575   2002 Fixed wing multi engine EMBRAER          EMB-1~       2    55    NA Turbo~
## 6 N105UW   1999 Fixed wing multi engine AIRBUS INDUSTRIE A320-~       2   182    NA Turbo~

# Comprobamos las variables de cada uno de los datasets que nos proporcionan mediante la instrucción "s
print(summary(flights))
```

```
##       year          month             day           dep_time    sched_dep_time
##  Min.   :2013   Min.   : 1.000   Min.   : 1.00   Min.   :   1   Min.   : 106
##  1st Qu.:2013   1st Qu.: 4.000   1st Qu.: 8.00   1st Qu.: 907   1st Qu.: 906
##  Median :2013   Median : 7.000   Median :16.00   Median :1401   Median :1359
##  Mean   :2013   Mean   : 6.549   Mean   :15.71   Mean   :1349   Mean   :1344
##  3rd Qu.:2013   3rd Qu.:10.000   3rd Qu.:23.00   3rd Qu.:1744   3rd Qu.:1729
##  Max.   :2013   Max.   :12.000   Max.   :31.00   Max.   :2400   Max.   :2359
##                                                  NA's   :8255
##    dep_delay          arr_time    sched_arr_time   arr_delay          carrier
##  Min.   : -43.00   Min.   :   1   Min.   :   1   Min.   : -86.000   Length:336776
##  1st Qu.:  -5.00   1st Qu.:1104   1st Qu.:1124   1st Qu.: -17.000   Class :character
##  Median :  -2.00   Median :1535   Median :1556   Median :  -5.000   Mode  :character
##  Mean   :  12.64   Mean   :1502   Mean   :1536   Mean   :   6.895
##  3rd Qu.:  11.00   3rd Qu.:1940   3rd Qu.:1945   3rd Qu.:  14.000
##  Max.   :1301.00   Max.   :2400   Max.   :2359   Max.   :1272.000
##  NA's   :8255      NA's   :8713                  NA's   :9430
##     flight       tailnum             origin              dest              air_time
##  Min.   :   1   Length:336776      Length:336776      Length:336776      Min.   : 20.0
##  1st Qu.: 553   Class :character   Class :character   Class :character   1st Qu.: 82.0
##  Median :1496   Mode  :character   Mode  :character   Mode  :character   Median :129.0
##  Mean   :1972                                                           Mean   :150.7
##  3rd Qu.:3465                                                           3rd Qu.:192.0
##  Max.   :8500                                                           Max.   :695.0
##                                                                         NA's   :9430
##     distance         hour           minute        time_hour
##  Min.   :  17   Min.   : 1.00   Min.   : 0.00   Min.   :2013-01-01 05:00:00.00
##  1st Qu.: 502   1st Qu.: 9.00   1st Qu.: 8.00   1st Qu.:2013-04-04 13:00:00.00
##  Median : 872   Median :13.00   Median :29.00   Median :2013-07-03 10:00:00.00
##  Mean   :1040   Mean   :13.18   Mean   :26.23   Mean   :2013-07-03 05:22:54.64
##  3rd Qu.:1389   3rd Qu.:17.00   3rd Qu.:44.00   3rd Qu.:2013-10-01 07:00:00.00
##  Max.   :4983   Max.   :23.00   Max.   :59.00   Max.   :2013-12-31 23:00:00.00
##
```

```
print(summary(airports))
```

```
##      faa                name               lat             lon
##  Length:1458        Length:1458        Min.   :19.72   Min.   :-176.65
##  Class :character   Class :character   1st Qu.:34.26   1st Qu.:-119.19
##  Mode  :character   Mode  :character   Median :40.09   Median : -94.66
##                                        Mean   :41.65   Mean   :-103.39
##                                        3rd Qu.:45.07   3rd Qu.: -82.52
##                                        Max.   :72.27   Max.   : 174.11
##       alt               tz              dst               tzone
##  Min.   : -54.00   Min.   :-10.000   Length:1458        Length:1458
##  1st Qu.:  70.25   1st Qu.: -8.000   Class :character   Class :character
##  Median : 473.00   Median : -6.000   Mode  :character   Mode  :character
##  Mean   :1001.42   Mean   : -6.519
##  3rd Qu.:1062.50   3rd Qu.: -5.000
##  Max.   :9078.00   Max.   :  8.000
```

```
print(summary(weather))
```

```
##     origin               year          month             day              hour
##  Length:26115       Min.   :2013   Min.   : 1.000   Min.   : 1.00   Min.   : 0.00
##  Class :character   1st Qu.:2013   1st Qu.: 4.000   1st Qu.: 8.00   1st Qu.: 6.00
```

```
##  Mode  :character   Median :2013   Median : 7.000   Median :16.00   Median :11.00
##                     Mean   :2013   Mean   : 6.504   Mean   :15.68   Mean   :11.49
##                     3rd Qu.:2013   3rd Qu.: 9.000   3rd Qu.:23.00   3rd Qu.:17.00
##                     Max.   :2013   Max.   :12.000   Max.   :31.00   Max.   :23.00
##
##       temp             dewp           humid           wind_dir       wind_speed
##  Min.   : 10.94   Min.   :-9.94   Min.   : 12.74   Min.   :  0.0   Min.   :   0.000
##  1st Qu.: 39.92   1st Qu.:26.06   1st Qu.: 47.05   1st Qu.:120.0   1st Qu.:   6.905
##  Median : 55.40   Median :42.08   Median : 61.79   Median :220.0   Median :  10.357
##  Mean   : 55.26   Mean   :41.44   Mean   : 62.53   Mean   :199.8   Mean   :  10.518
##  3rd Qu.: 69.98   3rd Qu.:57.92   3rd Qu.: 78.79   3rd Qu.:290.0   3rd Qu.:  13.809
##  Max.   :100.04   Max.   :78.08   Max.   :100.00   Max.   :360.0   Max.   :1048.361
##  NA's   :1        NA's   :1       NA's   :1        NA's   :460     NA's   :4
##    wind_gust         precip            pressure          visib
##  Min.   :16.11   Min.   :0.000000   Min.   : 983.8   Min.   : 0.000
##  1st Qu.:20.71   1st Qu.:0.000000   1st Qu.:1012.9   1st Qu.:10.000
##  Median :24.17   Median :0.000000   Median :1017.6   Median :10.000
##  Mean   :25.49   Mean   :0.004469   Mean   :1017.9   Mean   : 9.255
##  3rd Qu.:28.77   3rd Qu.:0.000000   3rd Qu.:1023.0   3rd Qu.:10.000
##  Max.   :66.75   Max.   :1.210000   Max.   :1042.1   Max.   :10.000
##  NA's   :20778                      NA's   :2729
##    time_hour
##  Min.   :2013-01-01 01:00:00.0
##  1st Qu.:2013-04-01 21:30:00.0
##  Median :2013-07-01 14:00:00.0
##  Mean   :2013-07-01 18:26:37.7
##  3rd Qu.:2013-09-30 13:00:00.0
##  Max.   :2013-12-30 18:00:00.0
##
```

```
print(summary(airlines))
```

```
##    carrier              name
##  Length:16          Length:16
##  Class :character   Class :character
##  Mode  :character   Mode  :character
```

```
print(summary(planes))
```

```
##    tailnum               year          type           manufacturer
##  Length:3322        Min.   :1956   Length:3322        Length:3322
##  Class :character   1st Qu.:1997   Class :character   Class :character
##  Mode  :character   Median :2001   Mode  :character   Mode  :character
##                     Mean   :2000
##                     3rd Qu.:2005
##                     Max.   :2013
##                     NA's   :70
##     model              engines          seats           speed          engine
##  Length:3322        Min.   :1.000   Min.   :  2.0   Min.   : 90.0   Length:3322
##  Class :character   1st Qu.:2.000   1st Qu.:140.0   1st Qu.:107.5   Class :character
##  Mode  :character   Median :2.000   Median :149.0   Median :162.0   Mode  :character
##                     Mean   :1.995   Mean   :154.3   Mean   :236.8
##                     3rd Qu.:2.000   3rd Qu.:182.0   3rd Qu.:432.0
##                     Max.   :4.000   Max.   :450.0   Max.   :432.0
##                                                     NA's   :3299
```

```r
# Simplificar los dataframes originales a 100 observaciones. Renombrarlos introduciendo la coletilla "_s

flights_simple <- head(flights,100)
airports_simple <- head(airports,100)
weather_simple <- head(weather,100)
airlines_simple <- head(airlines,100)
planes_simple <- head(planes,100)


# Selecciona los tipos de aerolinea ("carrier") mediante la instruccion "select" y el operador "unique"
airlines_simple %>% unique %>% select(carrier)
```

```
## # A tibble: 16 x 1
##    carrier
##    <chr>
##  1 9E
##  2 AA
##  3 AS
##  4 B6
##  5 DL
##  6 EV
##  7 F9
##  8 FL
##  9 HA
## 10 MQ
## 11 OO
## 12 UA
## 13 US
## 14 VX
## 15 WN
## 16 YV
```

```r
# Obtener la media y el maximo de asientos ("seats") que tienen los aviones. Utilizar el operador tuber
planes_simple %>% summarise(mean = mean(seats),max_engines = max(seats))
```

```
## # A tibble: 1 x 2
##    mean max_engines
##   <dbl>       <int>
## 1  105.         330
```

```r
# Ordenar los aviones por numero de motores ("engines") y numero de asientos ("seats").
result1 <- arrange(planes_simple,engines,seats)
print(result1)
```

```
## # A tibble: 100 x 9
##    tailnum  year type                    manufacturer model     engines seats speed engine
##    <chr>   <int> <chr>                   <chr>        <chr>       <int> <int> <int> <chr>
##  1 N10156   2004 Fixed wing multi engine EMBRAER      EMB-145XR       2    55    NA Turbo~
##  2 N10575   2002 Fixed wing multi engine EMBRAER      EMB-145LR       2    55    NA Turbo~
##  3 N11106   2002 Fixed wing multi engine EMBRAER      EMB-145XR       2    55    NA Turbo~
##  4 N11107   2002 Fixed wing multi engine EMBRAER      EMB-145XR       2    55    NA Turbo~
##  5 N11109   2002 Fixed wing multi engine EMBRAER      EMB-145XR       2    55    NA Turbo~
##  6 N11113   2002 Fixed wing multi engine EMBRAER      EMB-145XR       2    55    NA Turbo~
##  7 N11119   2002 Fixed wing multi engine EMBRAER      EMB-145XR       2    55    NA Turbo~
##  8 N11121   2003 Fixed wing multi engine EMBRAER      EMB-145XR       2    55    NA Turbo~
```

```
##  9 N11127   2003 Fixed wing multi engine EMBRAER      EMB-145XR       2    55    NA Turbo~
## 10 N11137   2003 Fixed wing multi engine EMBRAER      EMB-145XR       2    55    NA Turbo~
## # i 90 more rows

# Averigua que numero de cola comparten los dataframes "flights_simple" y "planes_simple" que has creado
# Obten su aerolinea ("carrier")
shared <- inner_join(flights_simple,planes_simple,by="tailnum") # -> N14228
shared_carrier <- shared$carrier
print(shared_carrier)

## [1] "EV"

# Cruzar los datos de vuelos ("flights") con los aviones ("planes") por el numero de cola ("tailnum") qu
# De esos obtener aquellos con 2 o mas motores.
# Finlmente obtener los distintos modelos de avión que satisfacen las premisas anteriores.
fp <- anti_join(planes_simple,flights_simple,by="tailnum")
engines_fp <- filter(fp,engines >= 2)
result2 <- unique(engines_fp$model) # No queremos los repetidos. Por lo tanto usamos "unique".
print(result2)

## [1] "EMB-145XR" "A320-214"  "EMB-145LR" "737-824"   "767-332"   "757-224"

# Crea una nueva variable que calcule el retraso total sumando los delays acumulados ("dep_delay") y ("
# Almacena el dataframe resultante en "flights_total".
flights_total <- mutate(flights_simple,total_delay=dep_delay+arr_delay)

# En base a la variable anteriormente obtenida, devuelve los aviones que han llegado con antelacion a su
filter(flights_total,total_delay < 0)

## # A tibble: 57 x 20
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time arr_delay
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>     <dbl>
##  1  2013     1     1      544            545        -1     1004           1022       -18
##  2  2013     1     1      554            600        -6      812            837       -25
##  3  2013     1     1      557            600        -3      709            723       -14
##  4  2013     1     1      557            600        -3      838            846        -8
##  5  2013     1     1      558            600        -2      849            851        -2
##  6  2013     1     1      558            600        -2      853            856        -3
##  7  2013     1     1      558            600        -2      923            937       -14
##  8  2013     1     1      559            559         0      702            706        -4
##  9  2013     1     1      559            600        -1      854            902        -8
## 10  2013     1     1      600            600         0      851            858        -7
## # i 47 more rows
## # i 11 more variables: carrier <chr>, flight <int>, tailnum <chr>, origin <chr>,
## #   dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>,
## #   time_hour <dttm>, total_delay <dbl>
```

Información de la sesión de R (incluyendo información sobre el sistema operativo, la versión de R y los paquetes usados):

```
sessionInfo()

## R version 4.3.1 (2023-06-16)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 20.04.6 LTS
```

```
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/atlas/libblas.so.3.10.3
## LAPACK: /usr/lib/x86_64-linux-gnu/atlas/liblapack.so.3.10.3;  LAPACK version 3.9.0
##
## locale:
##  [1] LC_CTYPE=es_ES.UTF-8       LC_NUMERIC=C               LC_TIME=es_ES.UTF-8
##  [4] LC_COLLATE=es_ES.UTF-8     LC_MONETARY=es_ES.UTF-8    LC_MESSAGES=es_ES.UTF-8
##  [7] LC_PAPER=es_ES.UTF-8       LC_NAME=C                  LC_ADDRESS=C
## [10] LC_TELEPHONE=C             LC_MEASUREMENT=es_ES.UTF-8 LC_IDENTIFICATION=C
##
## time zone: Europe/Madrid
## tzcode source: system (glibc)
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] nycflights13_1.0.2 dplyr_1.1.3        tidyr_1.3.0        knitr_1.44
## [5] factoextra_1.0.7   ggplot2_3.4.3      arules_1.7-6       Matrix_1.6-1.1
##
## loaded via a namespace (and not attached):
##  [1] gtable_0.3.4         xfun_0.40            recipes_1.0.8        ggrepel_0.9.3
##  [5] lattice_0.21-9       vctrs_0.6.3          tools_4.3.1          generics_0.1.3
##  [9] stats4_4.3.1         parallel_4.3.1       tibble_3.2.1         fansi_1.0.5
## [13] highr_0.10           pkgconfig_2.0.3      ModelMetrics_1.2.2.2 data.table_1.14.8
## [17] lifecycle_1.0.3      farver_2.1.1         compiler_4.3.1       stringr_1.5.0
## [21] tinytex_0.47         munsell_0.5.0        codetools_0.2-19     DALEX_2.4.3
## [25] htmltools_0.5.6.1    class_7.3-22         yaml_2.3.7           prodlim_2023.08.28
## [29] pillar_1.9.0         MASS_7.3-60          gower_1.0.1          iterators_1.0.14
## [33] rpart_4.1.19         foreach_1.5.2        nlme_3.1-163         parallelly_1.36.0
## [37] lava_1.7.2.1         tidyselect_1.2.0     digest_0.6.33        stringi_1.7.12
## [41] future_1.33.0        reshape2_1.4.4       purrr_1.0.2          listenv_0.9.0
## [45] labeling_0.4.3       splines_4.3.1        cowplot_1.1.1        fastmap_1.1.1
## [49] grid_4.3.1           colorspace_2.1-0     cli_3.6.1            magrittr_2.0.3
## [53] survival_3.5-7       utf8_1.2.3           future.apply_1.11.0  withr_2.5.1
## [57] scales_1.2.1         xgboost_1.7.5.1      lubridate_1.9.3      timechange_0.2.0
## [61] rmarkdown_2.25       globals_0.16.2       nnet_7.3-19          timeDate_4022.108
## [65] evaluate_0.22        hardhat_1.3.0        caret_6.0-94         rlang_1.1.1
## [69] Rcpp_1.0.11          glue_1.6.2           pROC_1.18.4          ipred_0.9-14
## [73] rstudioapi_0.15.0    jsonlite_1.8.7       R6_2.5.1             plyr_1.8.9
```

```
Sys.time()
```

```
## [1] "2023-10-31 22:29:49 CET"
```

Aprendizaje Automático I
Grado en Ciencia e Ingeniería de Datos
Universidad Rey Juan Carlos

Ejercicio propuesto: City Bike NYC

DSLab

diciembre, 2023

Introducción

El sistema de uso compartido de bicicletas en la ciudad de Nueva York (EE.UU.) publica diariamente gran cantidad de datos de actividad sobre su uso.

Estos datos han dado lugar, como no, a algunos análisis sobre la evolución de este servicio y posibles factores que puedan influenciar su uso. En esta práctica vamos a proponer el análisis de datos resumen diarios sobre la utilización de este servicio entre julio de 2013 y noviembre de 2015.

La filosofía de esta práctica es fomentar que consultéis la documentación en línea tanto de Pandas como de Seaborn, para así familiarizaros más con los diferentes métodos disponibles para resolver los ejercicios propuestos. En cada pregunta, se ofrecen consejos sobre partes relevantes de esta documentación relacionadas con las tareas que se piden.

Descripción de variables

El archivo de datos que vamos a utilizar puede obtenerse de esta url. Se trata de un fichero en formato CSV, que se ha creado mezclando datos del City Bike System con datos de la National Oceanic and Atmospheric Administration (NOAA), sobre NYC. El fichero cuenta

con las siguientes columnas:

- `date`: fecha del dato, en formato YYYY-MM-DD.
- `trips`: entero positivo, número total de viajes acumulados ese día.
- `precipitation`: entero positivo, cantidad de lluvia total registrada ese día (pulgadas).
- `snow_depth`: entero positivo, altura de nieve (pulgadas).
- `snowfall`: entero positivo, registro de precipitación en forma de nieve (pulgadas).
- `max_temperature`: entero, temperatura máxima registrada (ºF).
- `min_temperature`: entero, temperatura mínima registrada (ºF).
- `average_wind_speed`: entero, velocidad promedio del viento (MPH, millas por hora).
- `dow`: [0, 7]; código de día de la semana, 0 corresponde al domingo.
- `year`: Año del registro.
- `month`: Mes del registro.
- `holiday`: Valor lógico, indica si esa fecha es festivo (TRUE) o no (FALSE).
- `stations_in_service`: Número de estaciones para tomar o dejar bicicletas que estaban en servicio ese día.
- `weekday`: Valor lógico, indica si esa fecha corresponde a un día entre semana (de lunes a viernes, ambos inclusive).
- `weekday_non_holiday`: Valor lógico, indica si la fecha corresponde a un día entre semana festivo.

Los datos están tomados con frecuencia diaria (filas del archivo).

Ejercicio 1

Genera una tabla con valores estadísticos resumen para las variables cuantitativas de este conjunto de datos.

Ejercicio 2

Crea un gráfico que represente la evolución del número total de viajes en bicicleta registrados en el sistema cada mes.

A continuación, genera otro gráfico con la evolución de la media mensual de temperaturas máximas y mínimas.

¿Se pueden observar patrones estacionales o algún tipo de relación entre ambas variables?

Ejercicio 3

Representa un gráfico con dos paneles, en el que cada panel muestre el histograma y función de densidad de probabilidad del número total de viajes diarios realizados. El panel izquierdo mostrará la distribución del total de viajes diarios en dias no festivos y el panel derecho mostrará la misma distribución pero para días festivos.

Ejercicio 4

Calcula cual es, en promedio el día de la semana en el que más viajes en bicicleta se realizan y el día que menos viajes registra, usando toda la serie de valores. Si es posible, intenta visualizar estos datos por paneles para mostrar tus conclusiones.

Aprendizaje Automático I
Grado en Ciencia e Ingeniería de Datos
Universidad Rey Juan Carlos

Ejercicio: Componentes Principales

DSLab

diciembre, 2023

En este ejercicio vamos a trabajar con el conjunto de datos (Boston Housing Price) de la librería `mlbench`. El conjunto de datos consta de 506 observaciones de 14 atributos. El valor medio del precio de la vivienda en miles de dólares, denominado MEDV, es la variable de interés. A continuación se ofrece una breve descripción de cada característica:

- CRIM - tasa de delincuencia per cápita por ciudad
- ZN: proporción de suelo residencial con parcelas de más de 25.000 pies cuadrados.
- INDUS - proporción de acres comerciales no minoristas por ciudad
- CHAS - Variable ficticia del río Charles (1 si la zona linda con el río; 0 en caso contrario)
- NOX - concentración de óxidos nítricos (partes por 10 millones)
- RM - número medio de habitaciones por vivienda
- AGE - proporción de unidades ocupadas por sus propietarios construidas antes de 1940
- DIS - distancias ponderadas a cinco centros de empleo de Boston
- RAD - índice de accesibilidad a autopistas radiales
- TAX - tipo del impuesto sobre bienes inmuebles por cada 10.000 dólares
- PTRATIO - ratio alumnos-profesor por ciudad
- B - 1000(Bk - 0,63)^2 donde Bk es la proporción de negros por ciudad
- LSTAT - % de estatus inferior de la población
- MEDV - Valor medio de las viviendas ocupadas por sus propietarios en miles de $.

1. Carga los datos y explora su contenido.
2. Realiza un Análisis de Componentes Principales.
3. Estudia el efecto de escalar, o no, las variables.
4. ¿Cuántas PCs deberías de conservar?
5. Realiza e interpreta un biplot. ¿Puedes averiguar dónde se situan las casas más caras?
6. Busca una interpretación a las dos primeras componentes principales.

December 15, 2023

Solución ejercicio: Componentes Principales

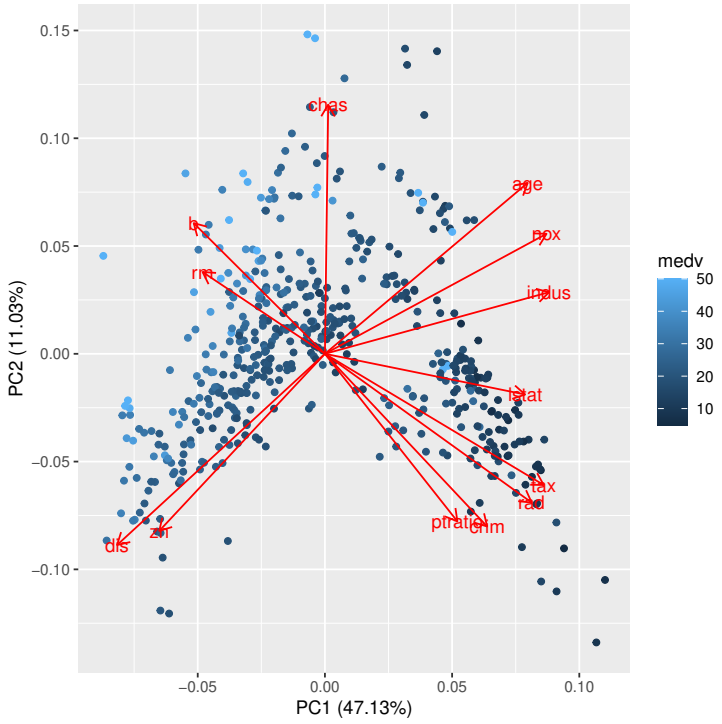
Los siguientes resultados han sido obtenidos con un script de R.

```r
# Librerías necesarias
library(mlbench)
library(dplyr)
library(ggfortify)


## Cargamos los datos
df=BostonHousing

summary(df)
```

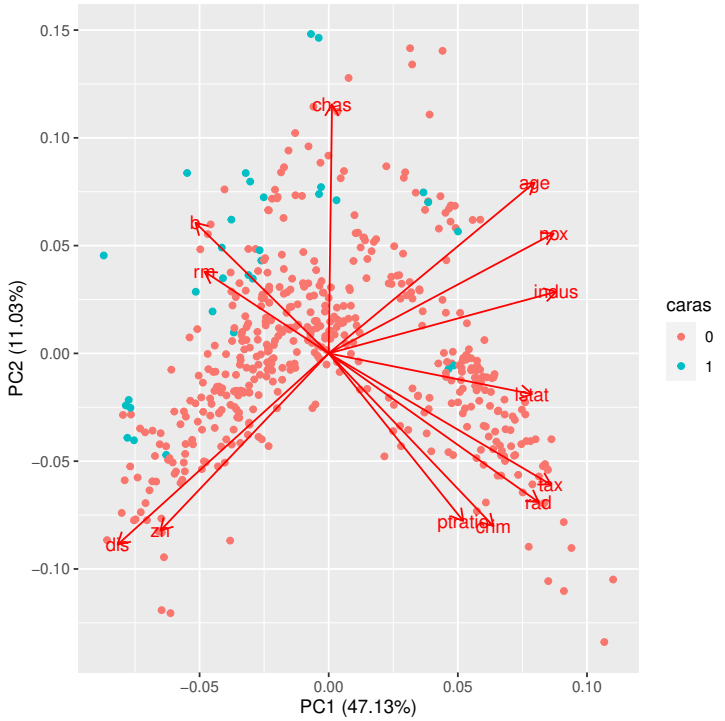
```
##       crim                zn              indus        chas         nox
##  Min.   : 0.00632   Min.   :  0.00   Min.   : 0.46   0:471   Min.   :0.3850
##  1st Qu.: 0.08205   1st Qu.:  0.00   1st Qu.: 5.19   1: 35   1st Qu.:0.4490
##  Median : 0.25651   Median :  0.00   Median : 9.69           Median :0.5380
##  Mean   : 3.61352   Mean   : 11.36   Mean   :11.14           Mean   :0.5547
##  3rd Qu.: 3.67708   3rd Qu.: 12.50   3rd Qu.:18.10           3rd Qu.:0.6240
##  Max.   :88.97620   Max.   :100.00   Max.   :27.74           Max.   :0.8710
##        rm              age              dis              rad              tax
##  Min.   :3.561   Min.   :  2.90   Min.   : 1.130   Min.   : 1.000   Min.   :187.0
##  1st Qu.:5.886   1st Qu.: 45.02   1st Qu.: 2.100   1st Qu.: 4.000   1st Qu.:279.0
##  Median :6.208   Median : 77.50   Median : 3.207   Median : 5.000   Median :330.0
##  Mean   :6.285   Mean   : 68.57   Mean   : 3.795   Mean   : 9.549   Mean   :408.2
##  3rd Qu.:6.623   3rd Qu.: 94.08   3rd Qu.: 5.188   3rd Qu.:24.000   3rd Qu.:666.0
##  Max.   :8.780   Max.   :100.00   Max.   :12.127   Max.   :24.000   Max.   :711.0
##     ptratio            b              lstat            medv
##  Min.   :12.60   Min.   :  0.32   Min.   : 1.73   Min.   : 5.00
##  1st Qu.:17.40   1st Qu.:375.38   1st Qu.: 6.95   1st Qu.:17.02
##  Median :19.05   Median :391.44   Median :11.36   Median :21.20
##  Mean   :18.46   Mean   :356.67   Mean   :12.65   Mean   :22.53
##  3rd Qu.:20.20   3rd Qu.:396.23   3rd Qu.:16.95   3rd Qu.:25.00
##  Max.   :22.00   Max.   :396.90   Max.   :37.97   Max.   :50.00
```

```r
# Convertimos la variable CHAS a numérica
df =
  df %>%
  mutate(chas=as.numeric(chas))

# PCA
df_pca <- prcomp(df[, -14], scale= TRUE)
summary(df_pca)
```

```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6     PC7     PC8
## Standard deviation     2.4752 1.1972 1.11473 0.92605 0.91368 0.81081 0.73168 0.62936
## Proportion of Variance 0.4713 0.1103 0.09559 0.06597 0.06422 0.05057 0.04118 0.03047
## Cumulative Proportion  0.4713 0.5816 0.67713 0.74310 0.80732 0.85789 0.89907 0.92954
##                           PC9    PC10    PC11    PC12    PC13
## Standard deviation     0.5263 0.46930 0.43129 0.41146 0.25201
## Proportion of Variance 0.0213 0.01694 0.01431 0.01302 0.00489
## Cumulative Proportion  0.9508 0.96778 0.98209 0.99511 1.00000

autoplot(df_pca,data=df,colour='medv',loadings=TRUE,loadings.label=TRUE)
```




```
# Las casas más caras parecen situarse, especialmente, en la esquina superior izquierda.
df =
  df %>%
  mutate(caras=as.factor(ifelse(medv>40,1,0)))

autoplot(df_pca,data=df,colour='caras',loadings=TRUE,loadings.label=TRUE)
```




```
# Las casas caras parecen estar asociadas a `chas`=1 y valores altos de "b"
# y "rm"

# Interpretación de las dos primeras componentes
df_pca$rotation[, 1:2]

##                 PC1         PC2
## crim     0.250951397 -0.31525237
## zn      -0.256314541 -0.32331290
## indus    0.346672065  0.11249291
## chas     0.005042434  0.45482914
## nox      0.342852313  0.21911553
## rm      -0.189242570  0.14933154
## age      0.313670596  0.31197778
## dis     -0.321543866 -0.34907000
## rad      0.319792768 -0.27152094
## tax      0.338469147 -0.23945365
## ptratio  0.204942258 -0.30589695
## b       -0.202972612  0.23855944
## lstat    0.309759840 -0.07432203

#La primera componente principal parece tener valores elevados en las variables
# `dis`, `zn`, 'b' y 'rm', frente a valores altos en todas las demas, a
# excepción de la variable chas. La segunda componente principal enfrenta
# observaciones con valores altos en `chas`, `age`, `nox`y `b`, frente a
# observaciones con valores altos en `crim`, `zn`, `dis` y `ptratio`.
```

Información de la sesión de R (incluyendo información sobre el sistema operativo, la versión de R y los paquetes usados):

```
sessionInfo()

## R version 4.3.1 (2023-06-16)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 20.04.6 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/atlas/libblas.so.3.10.3
## LAPACK: /usr/lib/x86_64-linux-gnu/atlas/liblapack.so.3.10.3;  LAPACK version 3.9.0
##
## locale:
##  [1] LC_CTYPE=es_ES.UTF-8       LC_NUMERIC=C               LC_TIME=es_ES.UTF-8
##  [4] LC_COLLATE=es_ES.UTF-8     LC_MONETARY=es_ES.UTF-8    LC_MESSAGES=es_ES.UTF-8
##  [7] LC_PAPER=es_ES.UTF-8       LC_NAME=C                  LC_ADDRESS=C
## [10] LC_TELEPHONE=C             LC_MEASUREMENT=es_ES.UTF-8 LC_IDENTIFICATION=C
##
## time zone: Europe/Madrid
## tzcode source: system (glibc)
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] ggfortify_0.4.16 factoextra_1.0.7 mlbench_2.1-3.1  readxl_1.4.3     caret_6.0-94
##  [6] lattice_0.21-9   ggplot2_3.4.3    rpart.plot_3.1.1 rpart_4.1.19     caTools_1.18.2
## [11] dplyr_1.1.3      ISLR2_1.3-2
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.2.0     timeDate_4022.108    farver_2.1.1         bitops_1.0-7
##  [5] fastmap_1.1.1        pROC_1.18.4          digest_0.6.33        timechange_0.2.0
##  [9] lifecycle_1.0.3      survival_3.5-7       magrittr_2.0.3       compiler_4.3.1
## [13] rlang_1.1.1          tools_4.3.1          utf8_1.2.3           yaml_2.3.7
## [17] data.table_1.14.8    knitr_1.44           labeling_0.4.3       plyr_1.8.9
## [21] withr_2.5.1          purrr_1.0.2          nnet_7.3-19          grid_4.3.1
## [25] stats4_4.3.1         fansi_1.0.5          e1071_1.7-13         colorspace_2.1-0
## [29] future_1.33.0        globals_0.16.2       scales_1.2.1         iterators_1.0.14
## [33] MASS_7.3-60          tinytex_0.47         cli_3.6.1            rmarkdown_2.25
## [37] generics_0.1.3       rstudioapi_0.15.0    future.apply_1.11.0  reshape2_1.4.4
## [41] tzdb_0.4.0           proxy_0.4-27         stringr_1.5.0        splines_4.3.1
## [45] parallel_4.3.1       cellranger_1.1.0     vctrs_0.6.3          hardhat_1.3.0
## [49] Matrix_1.6-1.1       hms_1.1.3            ggrepel_0.9.3        listenv_0.9.0
## [53] foreach_1.5.2        tidyr_1.3.0          gower_1.0.1          recipes_1.0.8
## [57] glue_1.6.2           parallelly_1.36.0    codetools_0.2-19     lubridate_1.9.3
## [61] stringi_1.7.12       gtable_0.3.4         munsell_0.5.0        tibble_3.2.1
## [65] pillar_1.9.0         htmltools_0.5.6.1    ipred_0.9-14         lava_1.7.2.1
## [69] R6_2.5.1             evaluate_0.22        readr_2.1.4          highr_0.10
## [73] class_7.3-22         Rcpp_1.0.11          gridExtra_2.3        nlme_3.1-163
## [77] prodlim_2023.08.28   xfun_0.40            pkgconfig_2.0.3      ModelMetrics_1.2.2.2

Sys.time()

## [1] "2023-11-01 22:05:09 CET"
```

Aprendizaje Automático I
Grado en Ciencia e Ingeniería de Datos
Universidad Rey Juan Carlos

Ejercicio: Aprendizaje No Supervisado

DSLab

diciembre, 2023

En este ejercicio vamos a trabajar con los datos del *"Dow Jones Index Data Set"* que podéis descargar aquí: DOW JONES INDEX. Se trata de datos semanales del Dow Jone Industrial Index.

Datos

En primer lugar descargamos y leemos los datos

```
djidata = read.table("./dow_jones_index/dow_jones_index.data",
    header = TRUE, sep = ",")
djidata = as.data.frame(djidata)
head(djidata)
```

```
##   quarter stock      date   open   high    low  close    volume
## 1       1    AA  1/7/2011 $15.82 $16.72 $15.78 $16.42 239655616
## 2       1    AA 1/14/2011 $16.71 $16.71 $15.64 $15.97 242963398
## 3       1    AA 1/21/2011 $16.19 $16.38 $15.60 $15.79 138428495
## 4       1    AA 1/28/2011 $15.87 $16.63 $15.82 $16.13 151379173
## 5       1    AA  2/4/2011 $16.18 $17.39 $16.18 $17.14 154387761
## 6       1    AA 2/11/2011 $17.33 $17.48 $16.97 $17.37 114691279
##   percent_change_price percent_change_volume_over_last_wk previous_weeks_volume
```

```
## 1             3.792670                                NA                  NA
## 2            -4.428490                          1.380223           239655616
## 3            -2.470660                        -43.024959           242963398
## 4             1.638310                          9.355500           138428495
## 5             5.933250                          1.987452           151379173
## 6             0.230814                        -25.712195           154387761
##   next_weeks_open next_weeks_close percent_change_next_weeks_price
## 1          $16.71           $15.97                       -4.428490
## 2          $16.19           $15.79                       -2.470660
## 3          $15.87           $16.13                        1.638310
## 4          $16.18           $17.14                        5.933250
## 5          $17.33           $17.37                        0.230814
## 6          $17.39           $17.28                       -0.632547
##   days_to_next_dividend percent_return_next_dividend
## 1                    26                      0.182704
## 2                    19                      0.187852
## 3                    12                      0.189994
## 4                     5                      0.185989
## 5                    97                      0.175029
## 6                    90                      0.172712
```

```
table(djidata$stock)
```

```
##
##   AA  AXP   BA  BAC  CAT CSCO  CVX   DD  DIS   GE   HD  HPQ  IBM INTC  JNJ  JPM
##   25   25   25   25   25   25   25   25   25   25   25   25   25   25   25   25
##   KO KRFT  MCD  MMM  MRK MSFT  PFE   PG    T  TRV  UTX   VZ  WMT  XOM
##   25   25   25   25   25   25   25   25   25   25   25   25   25   25
```

Cada fila corresponde a datos semanales de un valor bursatil. En este ejercicio vamos a trabajar con los datos correspondientes a la variable *close*, esto es, el valor de cada stock al cierre de la semana.

Necesitamos transformar la variable de interés como sigue:

```
djidata$close = as.numeric(sub("\\$", "", djidata$close))
```

1. Construir la matriz de series temporales

En primer lugar hemos de construir la matriz con las series que necesitamos. Necesitamos una matriz de series con las series por columnas para cada uno de los valores bursátiles.

```
stocks = unique(djidata[, "stock"])
n = dim(djidata[stocks == "AA", ])[1]
stocksdata = matrix(0, n, length(stocks))
for (i in 1:length(stocks)) stocksdata[, i] = djidata[djidata$stock ==
    stocks[i], "close"]
colnames(stocksdata) = stocks
stocksts1 = as.ts(stocksdata[1:12, ])
stocksts2 = as.ts(stocksdata[13:25, ])
stocksts = as.ts(stocksdata)
```

2. Representar las series con las que vamos a trabajar

3. Realizar un análisis cluster usando como variables de interés la media y desviación estándar de cada serie

¿Pueden identificarse valores atípicos?

¿Existe relacion entre las dos variables consideradas en el análisis? ¿Como interpretas este resultado?

4. Representar las series escaladas

5. Realizar un análisis Cluster para cada uno de los cuatrimestres

¿En cuantos grupos podemos dividir la muestra?

Representar graficamente la media de cada cluster para tratar de identificar el comportamiento medio de los valores en cada cluster.

6. Análisis Cluster para todo el periodo

Elegir una tecnica para determinar el mejor numero de clusters.

7. Representar graficamente la media de cada cluster

8. Localizar atípicos en los clusters

9. Repetir el análisis, para todo el periodo, empleando la distancia DTW.

10. Identificar las diferencias entre los dos análisis

Aprendizaje Automático I
Grado en Ciencia e Ingeniería de Datos
Universidad Rey Juan Carlos

Solución ejercicio: Aprendizaje No Supervisado

DSLab

diciembre, 2023

En este ejercicio vamos a trabajar con los datos del *"Dow Jones Index Data Set"* que podéis descargar en el siguiente enlace: DOW JONES INDEX. Se trata de datos semanales del Dow Jone Industrial Index.

Datos

En primer lugar descargamos los datos y leemos los datos

```
djidata = read.table("./dow_jones_index/dow_jones_index.data",
    header = TRUE, sep = ",")
djidata = as.data.frame(djidata)
head(djidata)
```

```
##   quarter stock      date   open   high    low  close    volume
## 1       1    AA  1/7/2011 $15.82 $16.72 $15.78 $16.42 239655616
## 2       1    AA 1/14/2011 $16.71 $16.71 $15.64 $15.97 242963398
## 3       1    AA 1/21/2011 $16.19 $16.38 $15.60 $15.79 138428495
## 4       1    AA 1/28/2011 $15.87 $16.63 $15.82 $16.13 151379173
## 5       1    AA  2/4/2011 $16.18 $17.39 $16.18 $17.14 154387761
## 6       1    AA 2/11/2011 $17.33 $17.48 $16.97 $17.37 114691279
##   percent_change_price percent_change_volume_over_last_wk previous_weeks_volume
```

```
## 1             3.792670                                NA                  NA
## 2            -4.428490                          1.380223           239655616
## 3            -2.470660                        -43.024959           242963398
## 4             1.638310                          9.355500           138428495
## 5             5.933250                          1.987452           151379173
## 6             0.230814                        -25.712195           154387761
##   next_weeks_open next_weeks_close percent_change_next_weeks_price
## 1          $16.71           $15.97                       -4.428490
## 2          $16.19           $15.79                       -2.470660
## 3          $15.87           $16.13                        1.638310
## 4          $16.18           $17.14                        5.933250
## 5          $17.33           $17.37                        0.230814
## 6          $17.39           $17.28                       -0.632547
##   days_to_next_dividend percent_return_next_dividend
## 1                    26                      0.182704
## 2                    19                      0.187852
## 3                    12                      0.189994
## 4                     5                      0.185989
## 5                    97                      0.175029
## 6                    90                      0.172712
```

```
table(djidata$stock)
```

```
##
##   AA  AXP   BA  BAC  CAT CSCO  CVX   DD  DIS   GE   HD  HPQ  IBM INTC  JNJ  JPM
##   25   25   25   25   25   25   25   25   25   25   25   25   25   25   25   25
##   KO KRFT  MCD  MMM  MRK MSFT  PFE   PG    T  TRV  UTX   VZ  WMT  XOM
##   25   25   25   25   25   25   25   25   25   25   25   25   25   25
```

Cada fila corresponde a datos semanales de un valor bursatil. En este ejercicio vamos a trabajar con los datos correspondientes a la variable *close*, esto es, el valor al cierre de la semana del stock.

Necesitamos transformar la variable de interés como sigue:

```
djidata$close = as.numeric(sub("\\$", "", djidata$close))
```

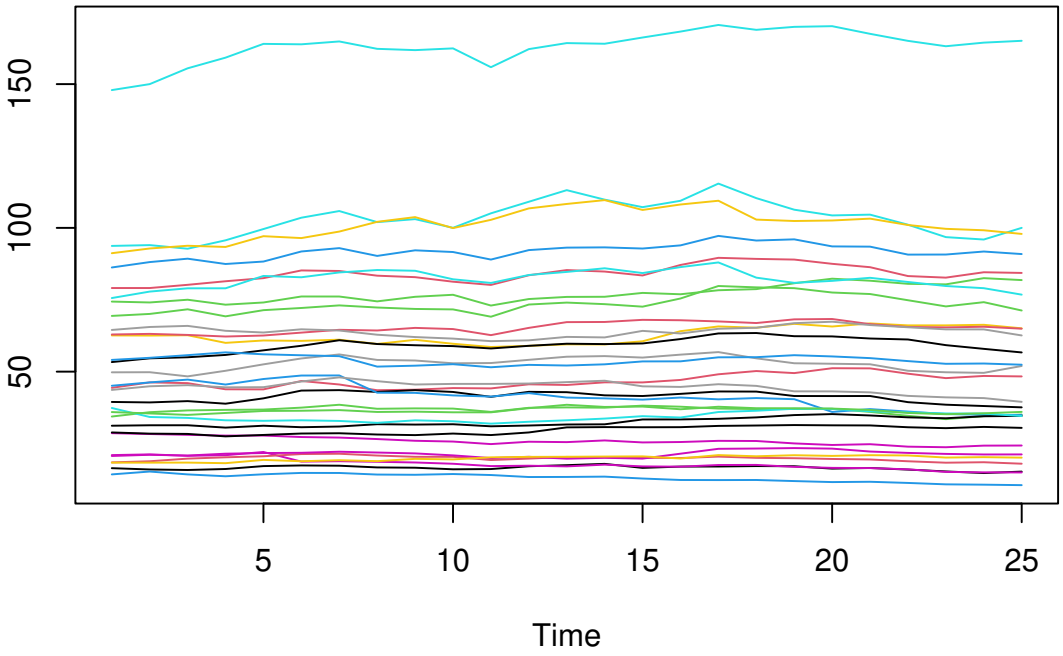
1. Construir la matriz de series temporales

En primer lugar hemos de construir la matriz con las series que necesitamos. Necesitamos una matriz de series con las series por columnas para cada uno de los valores bursátiles.

```
stocks = unique(djidata[, "stock"])
n = dim(djidata[stocks == "AA", ])[1]
stocksdata = matrix(0, n, length(stocks))
for (i in 1:length(stocks)) stocksdata[, i] = djidata[djidata$stock ==
    stocks[i], "close"]
colnames(stocksdata) = stocks
stocksts1 = as.ts(stocksdata[1:12, ])
stocksts2 = as.ts(stocksdata[13:25, ])
stocksts = as.ts(stocksdata)
```

2. Representar las series con las que vamos a trabajar

```
ts.plot(stocksts, col = seq(1:25))
```



A la vista de este gráfico, podríamos realizar nuestro análisis basándono, únicamente, en dos caractarísticas de las series: su media y desviación típica.

3. Realizar un análisis cluster usando como variables de interés la media y desviación estándar de cada serie

```
require(dplyr)
```

```
## Loading required package: dplyr
```

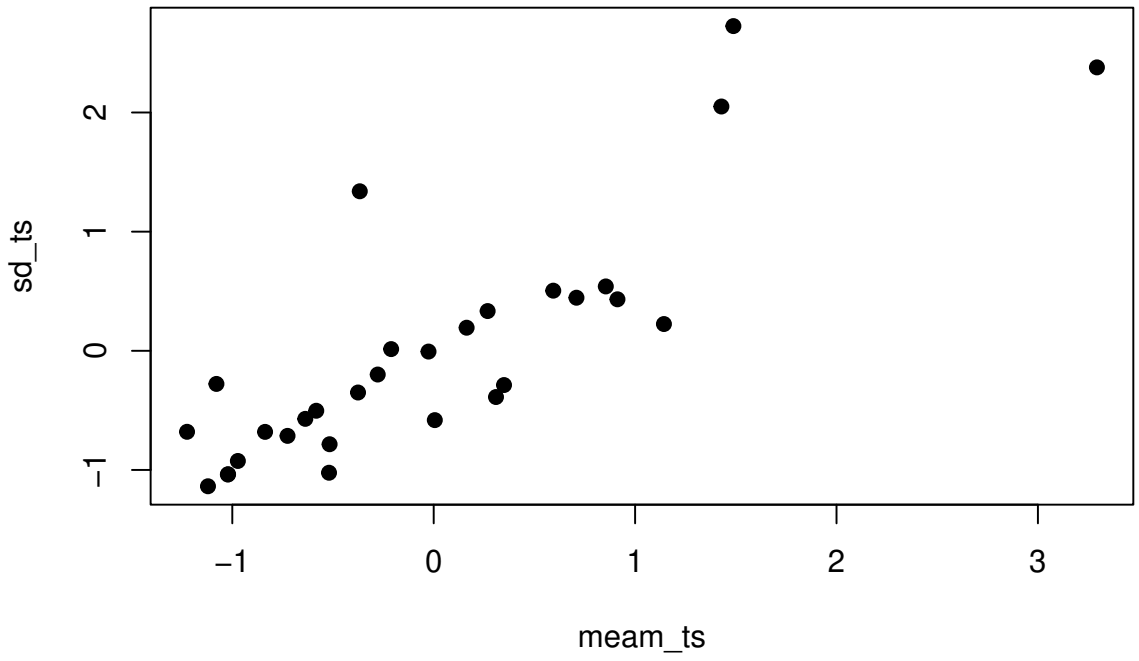
```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
```

```
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

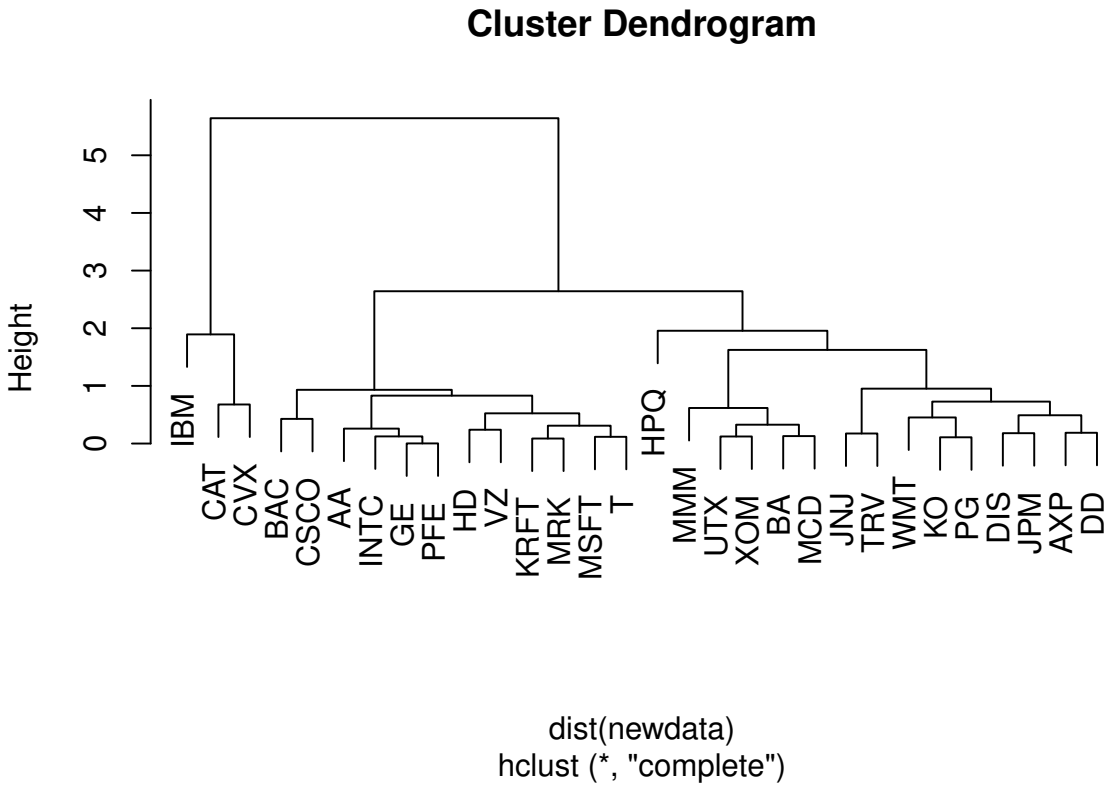
```
newdata = djidata %>%
    group_by(stock) %>%
    summarise(meam_ts = mean(close), sd_ts = sd(close))
newdata = as.data.frame(newdata)
row.names(newdata) = newdata[, 1]
newdata = scale(newdata[, -1])
plot(newdata, pch = 19)
```



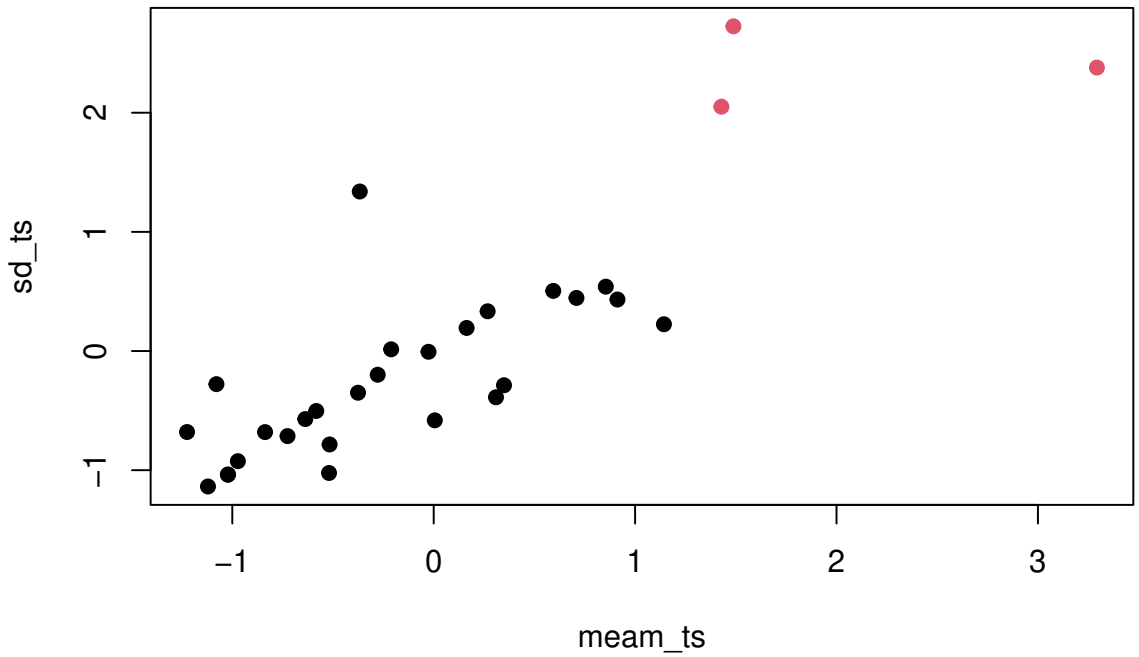

```
require(cluster)
```

```
## Loading required package: cluster
```

```
djicluster = hclust(dist(newdata))
plot(djicluster)
```




```
djicluster2 = cutree(djicluster, k = 2)
plot(newdata, pch = 19, col = djicluster2)
```



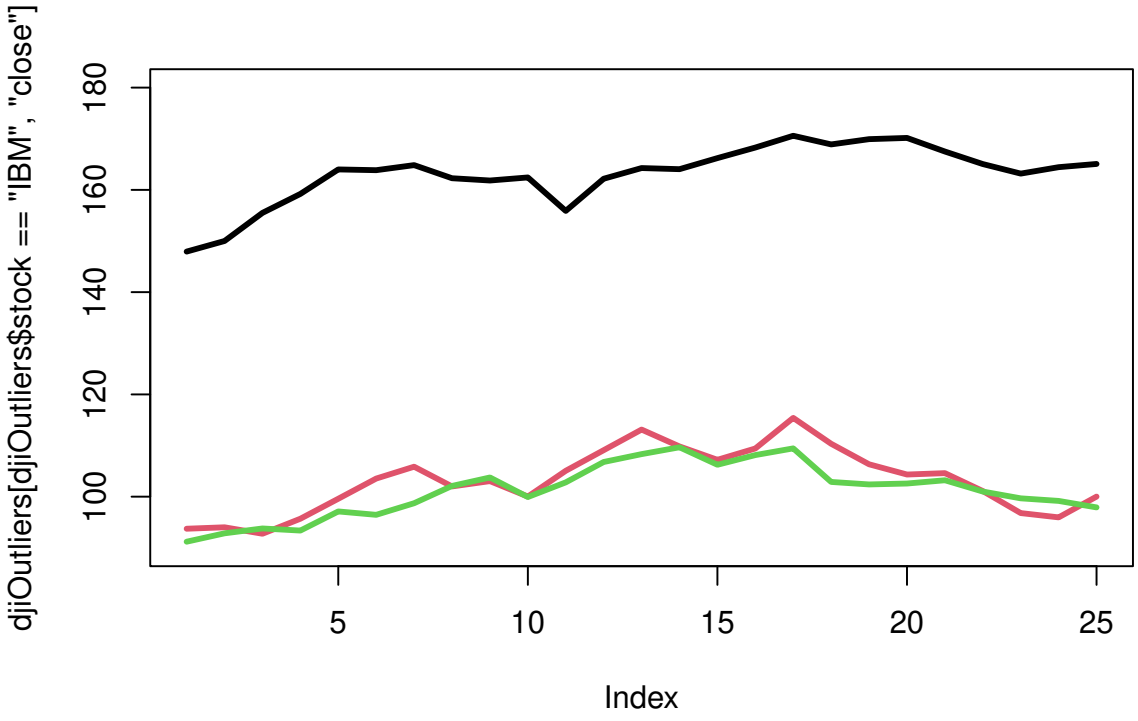
```
djicluster2
```

```
##   AA  AXP   BA  BAC  CAT CSCO  CVX   DD  DIS   GE   HD  HPQ  IBM INTC  JNJ  JPM
##    1    1    1    1    2    1    2    1    1    1    1    1    2    1    1    1
##   KO KRFT  MCD  MMM  MRK MSFT  PFE   PG    T  TRV  UTX   VZ  WMT  XOM
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1
```

¿Pueden identificarse valores atípicos?

Todos los atipicos aparecen en el grupo 2.

```
cselect = c("CAT", "CVX", "IBM")
djiOutliers = djidata %>%
    filter(stock == "CAT" | stock == "CVX" | stock == "IBM")
plot(djiOutliers[djiOutliers$stock == "IBM", "close"], type = "l",
    lwd = 3, ylim = c(90, 180))
points(djiOutliers[djiOutliers$stock == "CAT", "close"], type = "l",
    lwd = 3, col = 2)
points(djiOutliers[djiOutliers$stock == "CVX", "close"], type = "l",
    lwd = 3, col = 3)
```



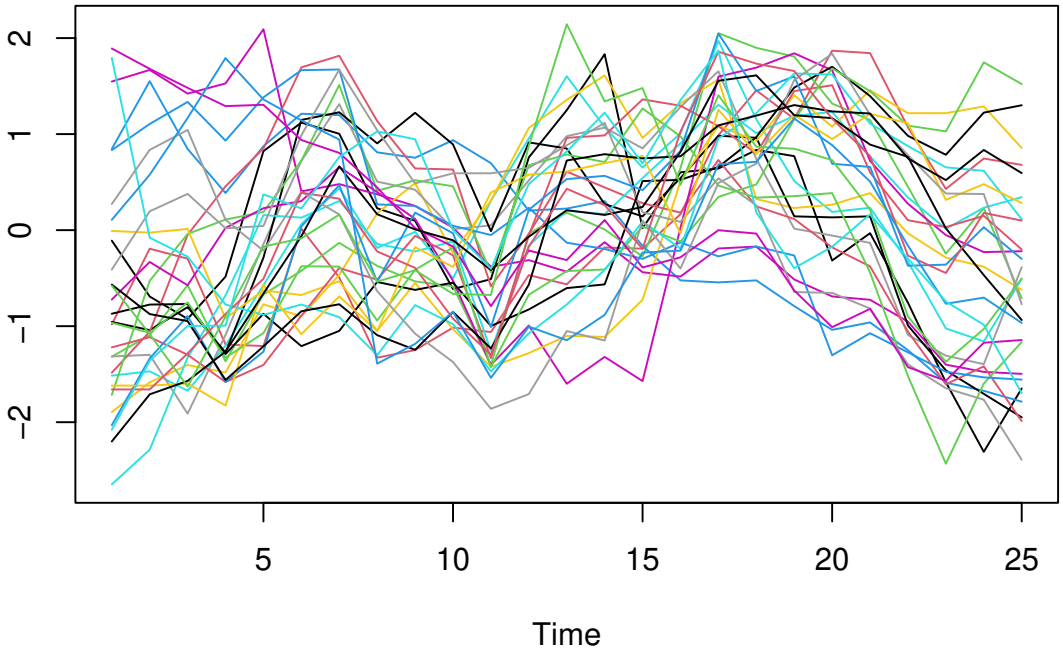
¿Existe relacion entre las dos variables consideradas en el analisis? ¿Como interpretas este resultado?

Efectivamente, existe una relacion lineal positiva entre media y desviacion tipica. Por

lo tanto, lo mas logico seria escalar los datos para que todos tengan la misma media y desviacion tipica.

4. Representar las series escaladas

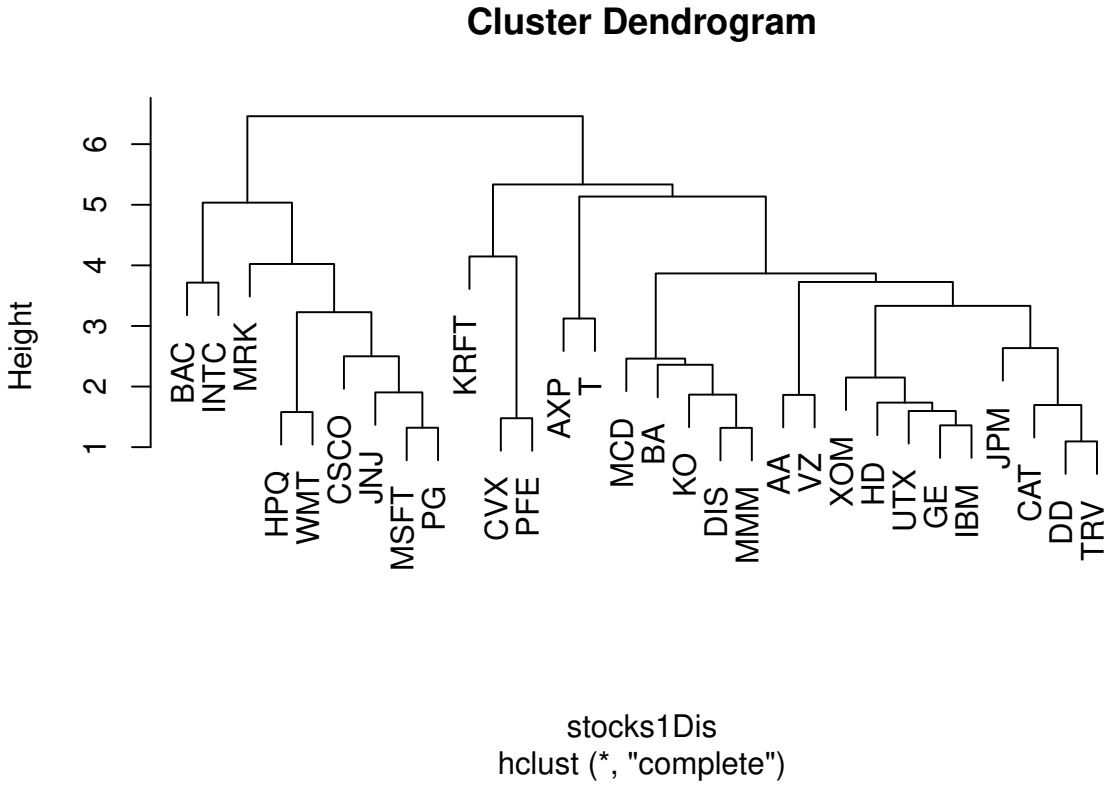
```
ts.plot(scale(stocksts), col = seq(1:25))
```



Una vez contruida la matriz con las series temporales, podemos pasar a analizar los datos. Separaremos los datos por cuatrimestres (*quarter*). Realizaremos un análisis cluster para cada uno de los cuatrimestres y otro empleando todo el periodo.

5. Análisis Cluster para cada uno de los cuatrimestres

```
# Usamos la distancia euclídea y un método jerárquico.
stocksts1Scaled = scale(stocksts1)
stocks1Dis = dist(t(stocksts1Scaled))
cluster1 = hclust(stocks1Dis)
plot(cluster1)
```



¿En cuantos grupos podemos dividir la muestra?

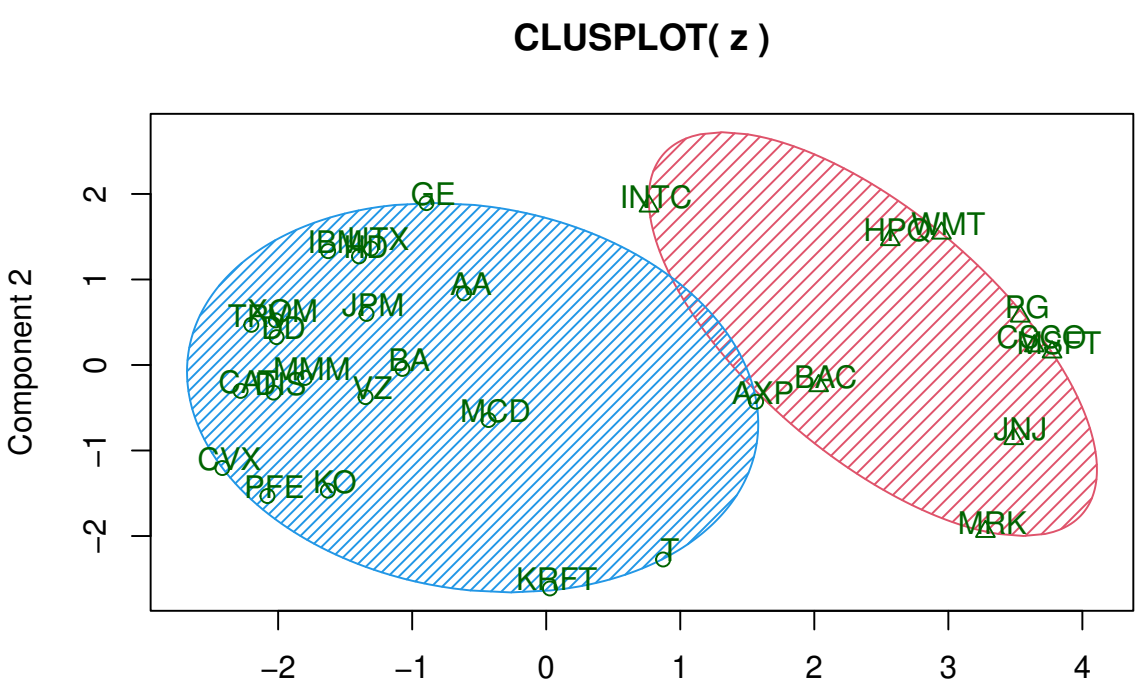
A la vista del dendograma intuimos 2 grupos

```
# z3=pam(t(stocksts1Scaled),2)
# z1=kmeans(t(stocksts1Scaled),2,nstart=25)
z1 = cutree(cluster1, 2)
require(useful)
```

```
## Loading required package: useful
```

```
## Loading required package: ggplot2
```

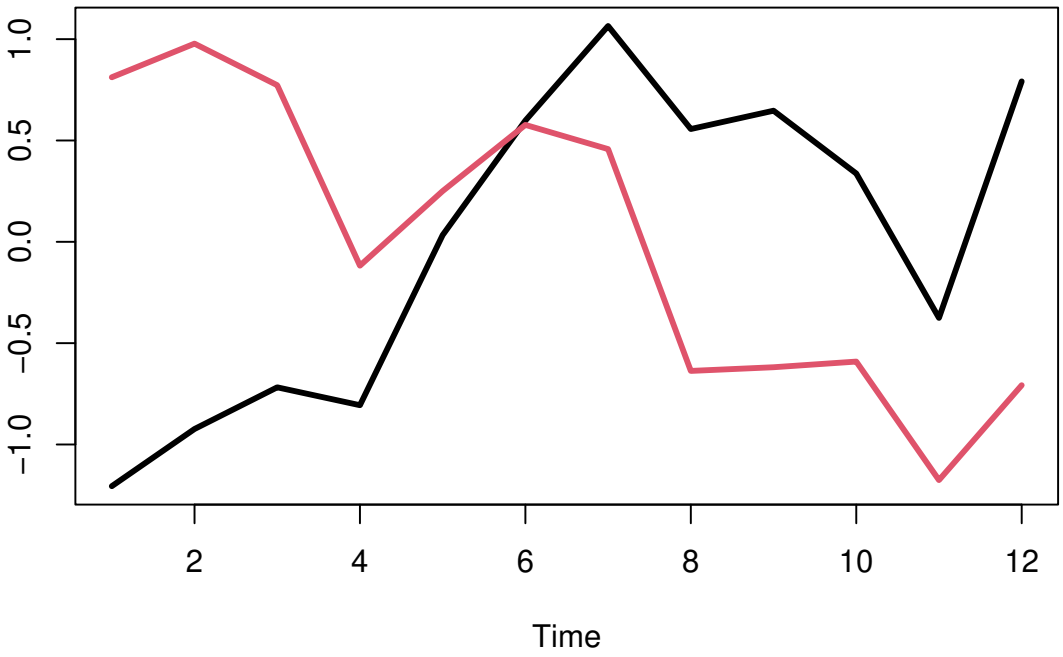
```
z = cmdscale(stocks1Dis)
clusplot(z, labels = 3, clus = z1, shade = TRUE, color = TRUE)
```



Representar graficamente la media de cada cluster para tratar de identificar el comportamiento medio de los valores en cada cluster.

Buscamos una representación media del comportamiento en cada cluster.

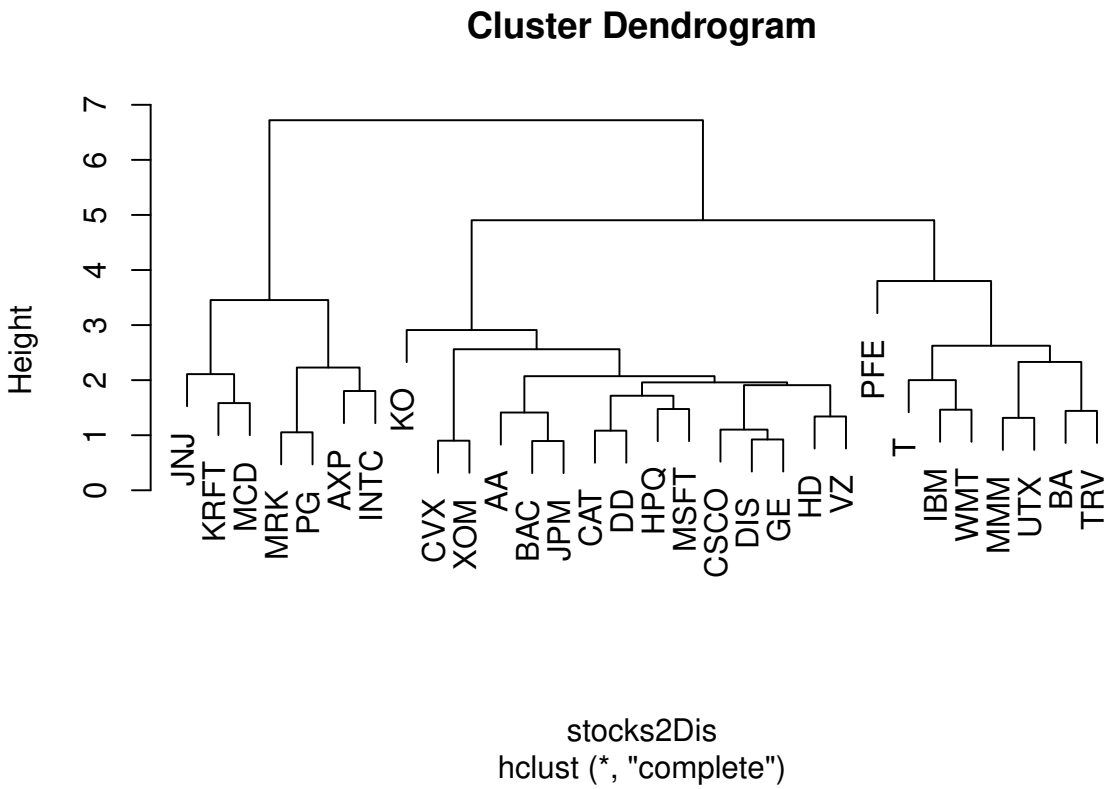
```
z1 = kmeans(t(stocksts1Scaled), 2, nstart = 25)
ts.plot(t(z1$centers), col = 1:2, lwd = 3)
```



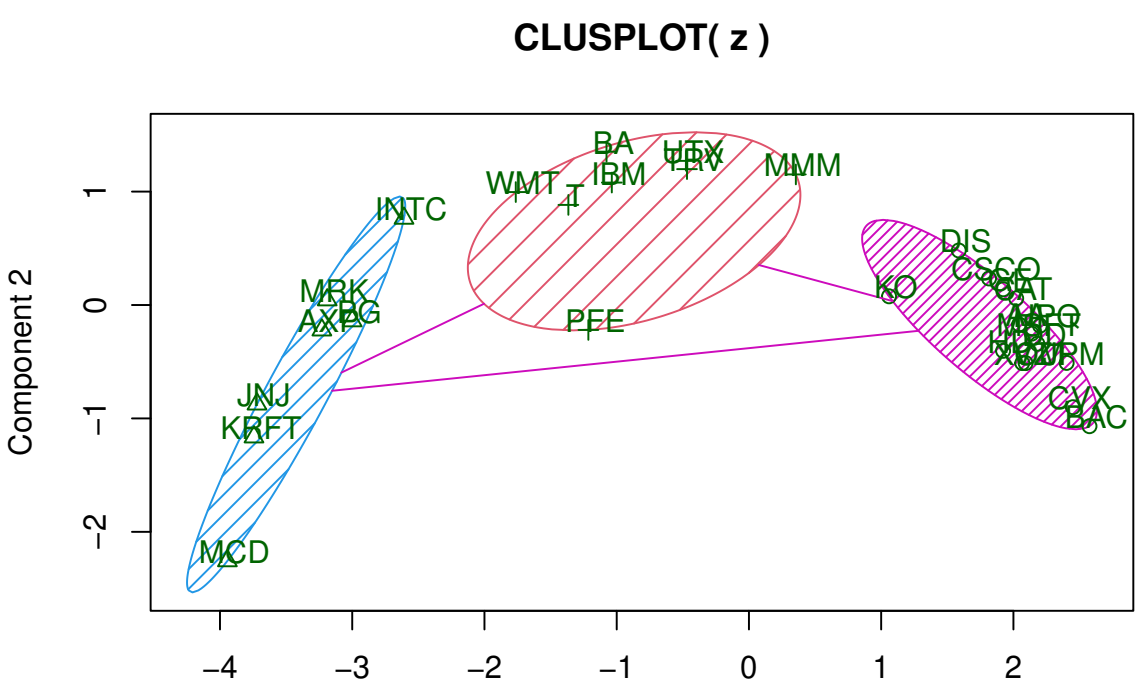
Podemos observar que el primer cluster corresponde a valores que crecen con el tiempo y el segundo cluster a valores que decrecen con el tiempo.

Pasamos a trabajar con el segundo cuatrimestre.

```
stocksts2Scaled = scale(stocksts2)
stocks2Dis = dist(t(stocksts2Scaled))
cluster2 = hclust(stocks2Dis)
plot(cluster2)
```

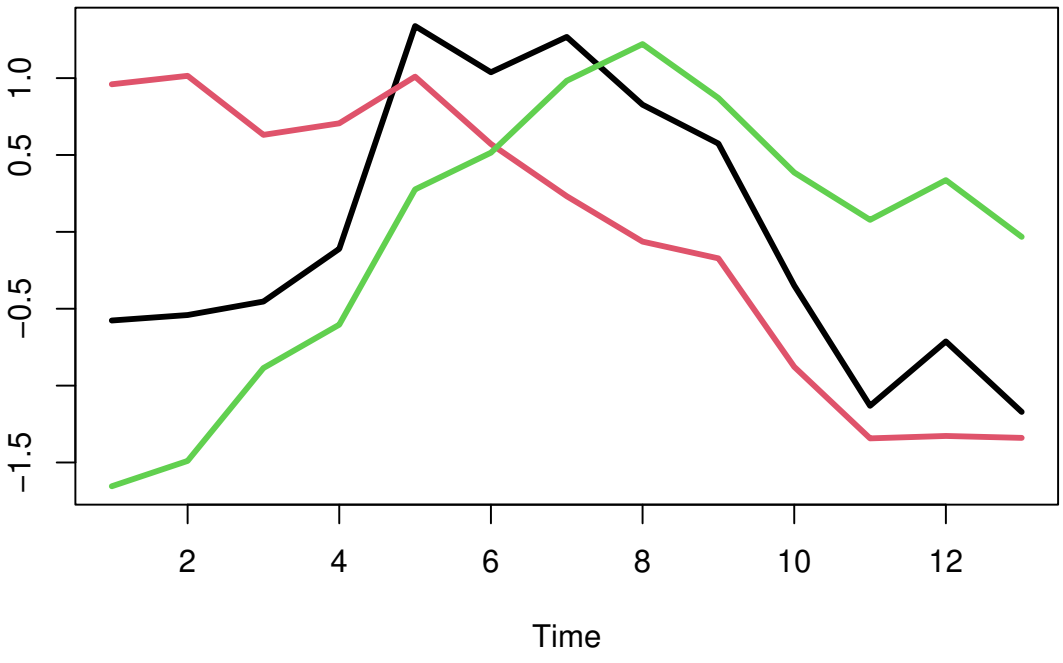



```
# A la vista del dendograma intuimos 3 grupos
z2 = cutree(cluster2, 3)
# z2=kmeans(t(stocksts2Scaled),3,nstart=25)
require(useful)
z = cmdscale(dist(t(stocksts2Scaled)))
clusplot(z, labels = 3, clus = z2, shade = TRUE, color = TRUE)
```



Buscamos una representación media del comportamiento en cada cluster.

```
z2 = kmeans(t(stocksts2Scaled), 3, nstart = 25)
ts.plot(t(z2$centers), col = 1:3, lwd = 3)
```



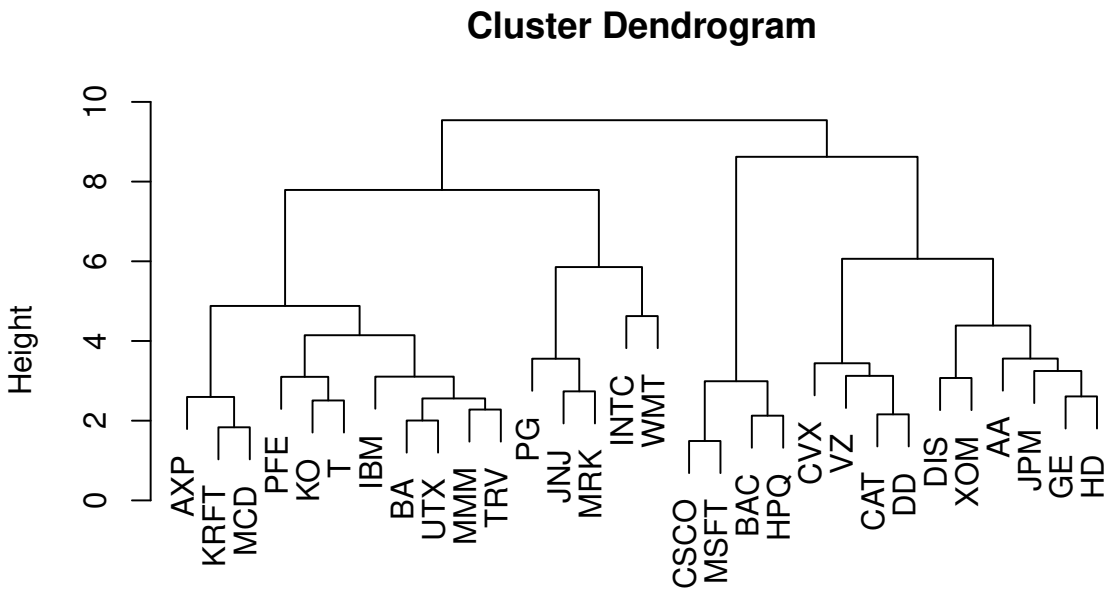
Podemos observar que el primer cluster corresponde a valores que decrecen con el

tiempo. El segundo cluster corresponde a valores que crecen, se mantienen más o menos constantes y luego decrecen. El tercer cluster corresponde a valores que crecen y luego decrecen.

6. Analisis Cluster para todo el periodo

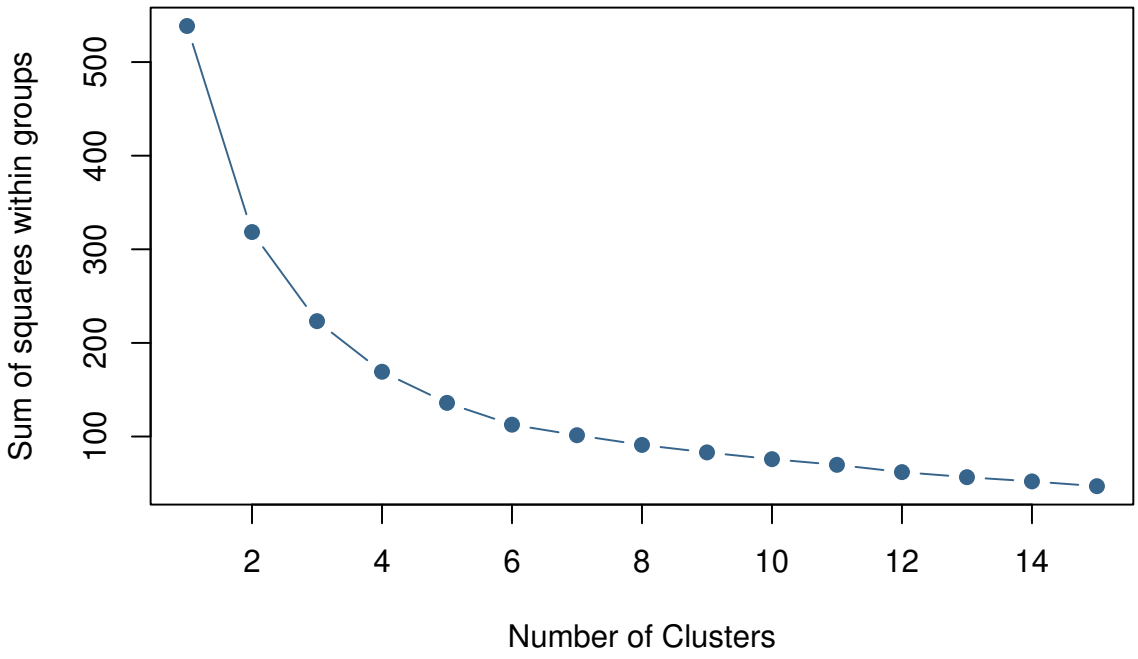
Trabajamos con los datos de todo el periodo analizado.

```
# todo el periodo
stockstsScaled = scale(stocksts)
stocksDis = dist(t(stockstsScaled))
clusterTotal = hclust(stocksDis)
plot(clusterTotal)
```

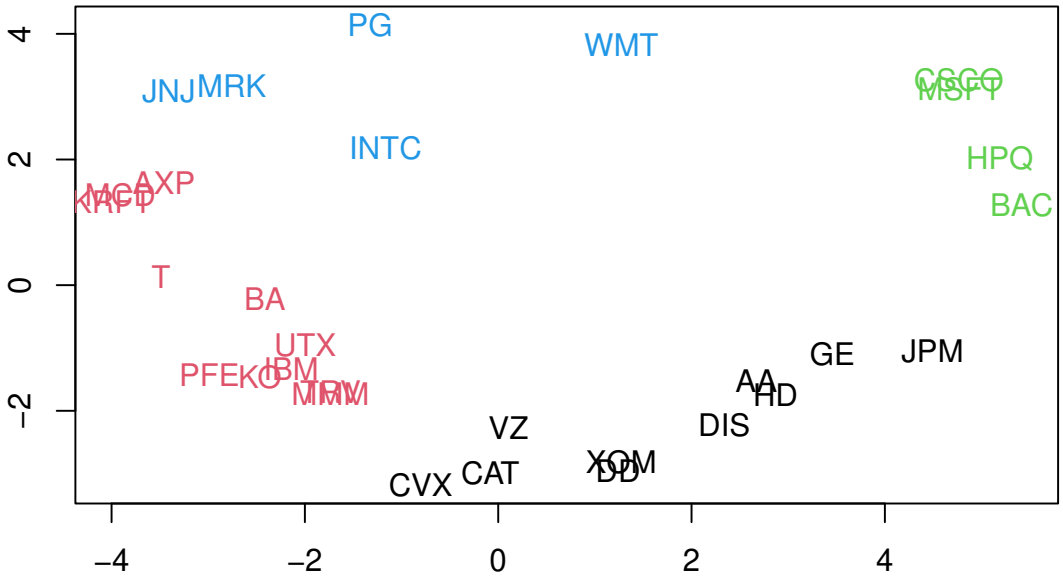


Elegir una tecnica para determinar el mejor numero de clusters.

```
SSW <- vector(mode = "numeric", length = 15)
SSW[1] <- (30 - 1) * sum(apply(t(stockstsScaled), 2, var))
for (i in 2:15) SSW[i] <- sum(kmeans(t(stockstsScaled), centers = i,
    nstart = 25)$withinss)
plot(1:15, SSW, type = "b", xlab = "Number of Clusters", ylab = "Sum of squares within g
    pch = 19, col = "steelblue4")
```

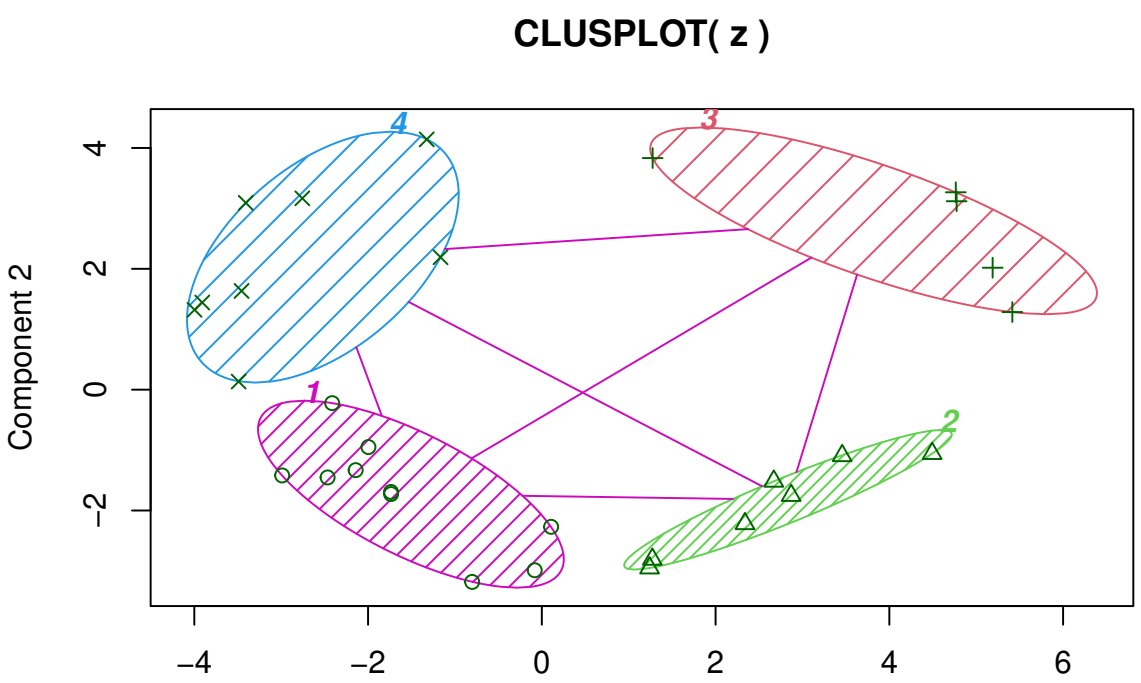



```
# A la vista del dendograma intuimos 4 grupos
z3 = cutree(clusterTotal, 4)
require(useful)
z = cmdscale(dist(t(stockstsScaled)))
plot(z[, 1], z[, 2], type = "n", xlab = "", ylab = "")
text(z[, 1], z[, 2], rownames(z), cex = 1, col = z3)
```

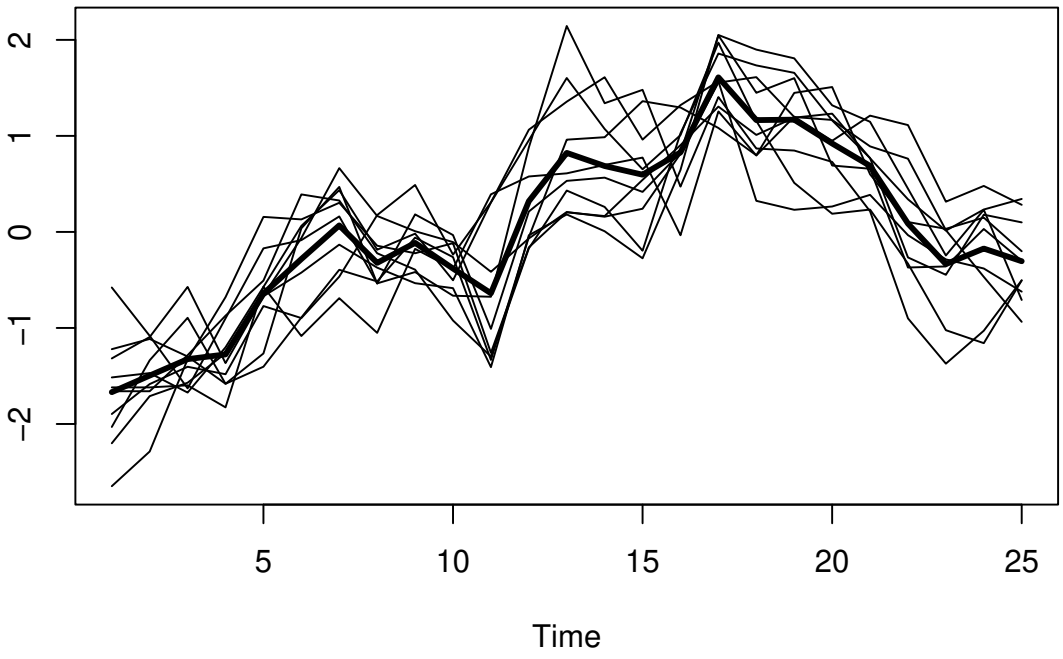


7. Representar graficamente la media de cada cluster Vamos a ver,las medias de cada uno de los clusters.

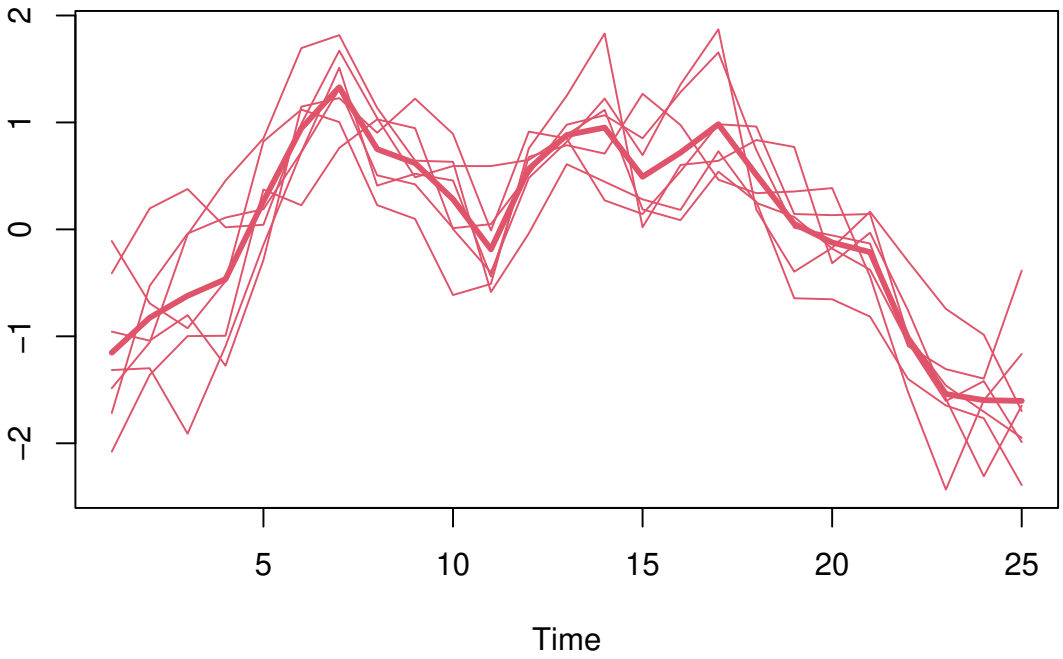
```
# A la vista del dendograma intuimos 3 grupos
z3 = kmeans(t(stockstsScaled), 4, nstart = 25)
require(useful)
clusplot(z, labels = 4, clus = z3$cluster, shade = TRUE, color = TRUE)
```



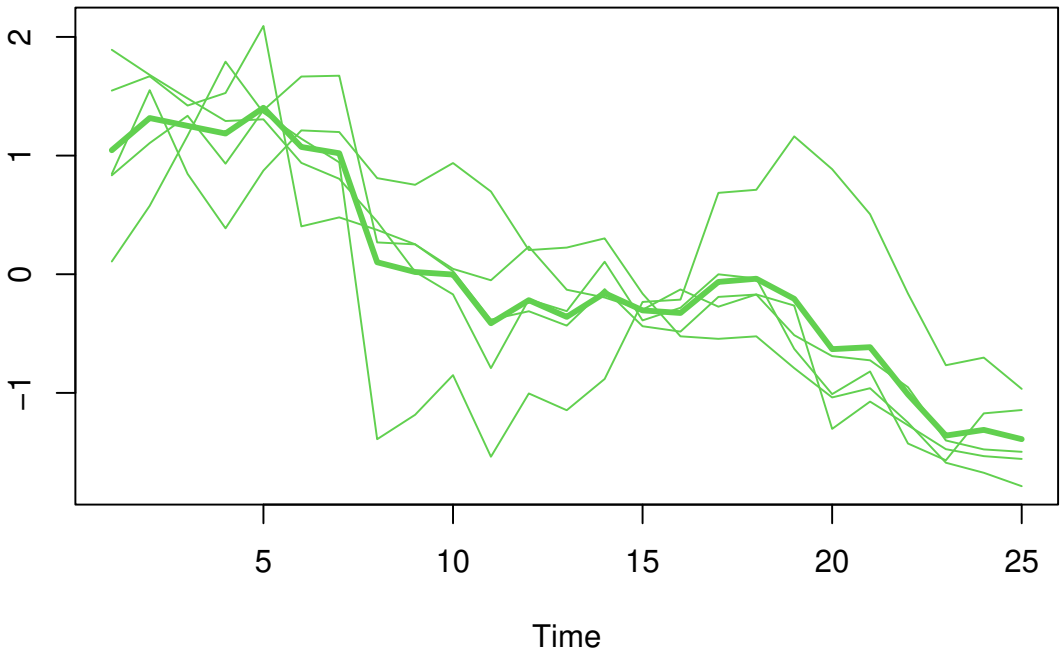

```
ts.plot(cbind((z3$centers[1, ]), stockstsScaled[, z3$cluster ==
    1]), lwd = c(3, rep(1, sum(z3$cluster == 1))), col = 1)
```



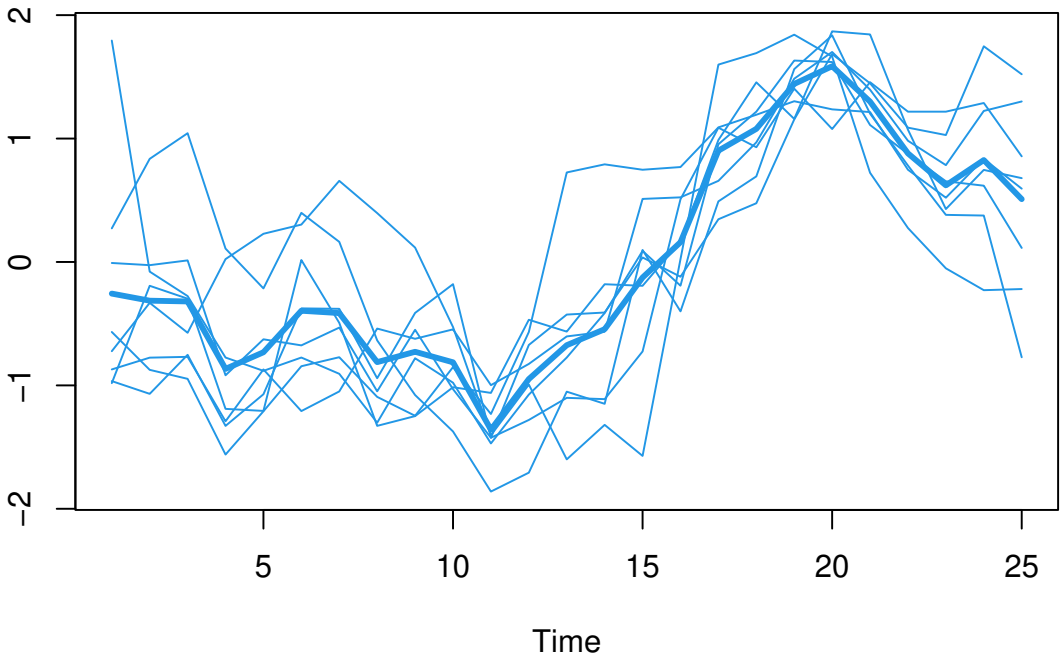

```
ts.plot(cbind((z3$centers[2, ]), stockstsScaled[, z3$cluster ==
    2]), lwd = c(3, rep(1, sum(z3$cluster == 2))), col = 2)
```




```
ts.plot(cbind((z3$centers[3, ]), stockstsScaled[, z3$cluster ==
    3]), lwd = c(3, rep(1, sum(z3$cluster == 3))), col = 3)
```



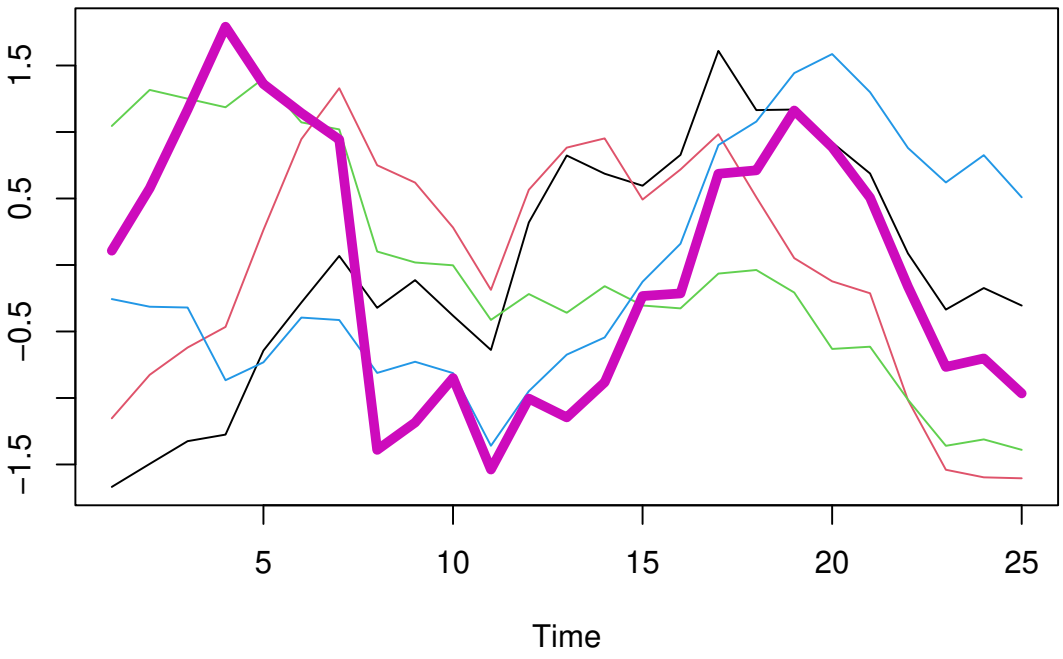

```
ts.plot(cbind((z3$centers[4, ]), stockstsScaled[, z3$cluster ==
    4]), lwd = c(3, rep(1, sum(z3$cluster == 4))), col = 4)
```



El primer cluster corresponde a valores decrecientes en el tiempo. El segundo cluster corresponde a valores que permanecen aproximadamente constantes hasta la mitad del periodo para crecer a partir de ese momento. El tercer cluster corresponde a valores crecientes en el tiempo. El cuarto cluster corresponde a valores que crecen, permanecen contantes, y decrecen.

8. Localizar atipicos en los clusters Observando detalladamente las series y los resultados obtenidos en este análisis podemos ver que la serie correspondiente a *WMT* presenta un comportamiento que no responde a ninguno de estos clusters.

```
ts.plot(cbind(t(z3$centers), stockstsScaled[, "WMT"]), col = c(1:4,
    6), lwd = c(rep(1, 4), 5))
```



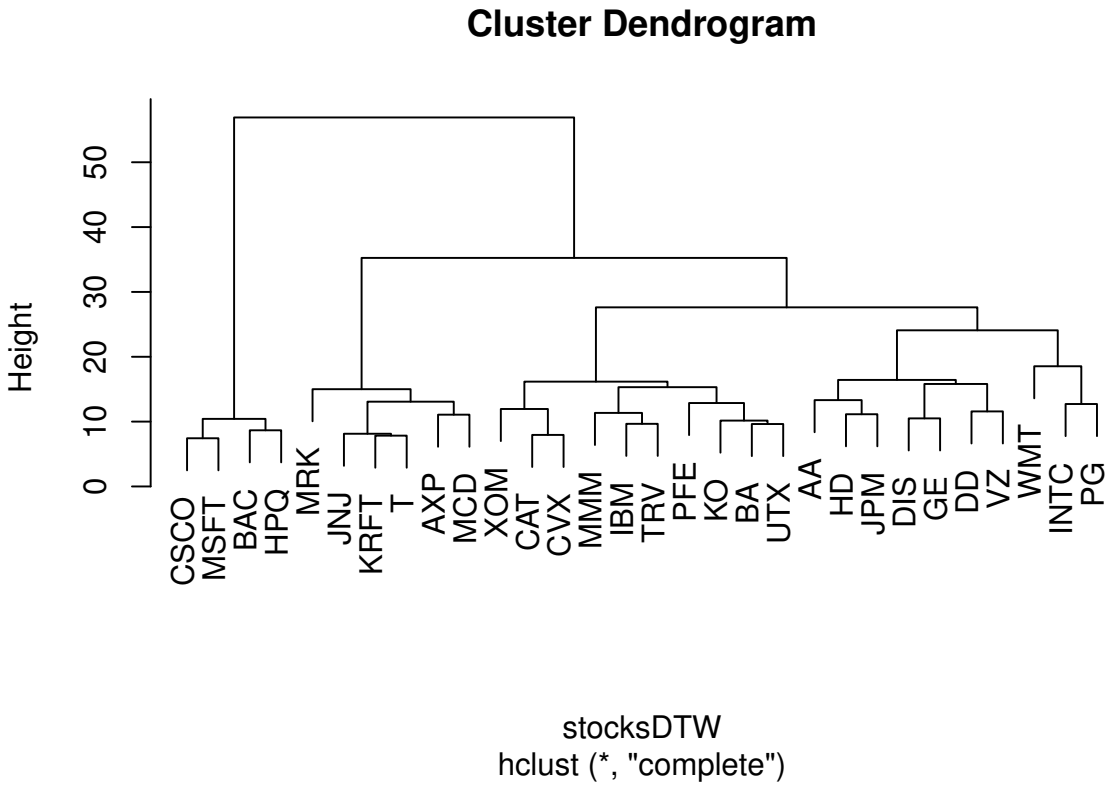
---

9. Repetir el análisis, para todo el periodo, empleando la distancia DTW.

```
require(dtw)
```

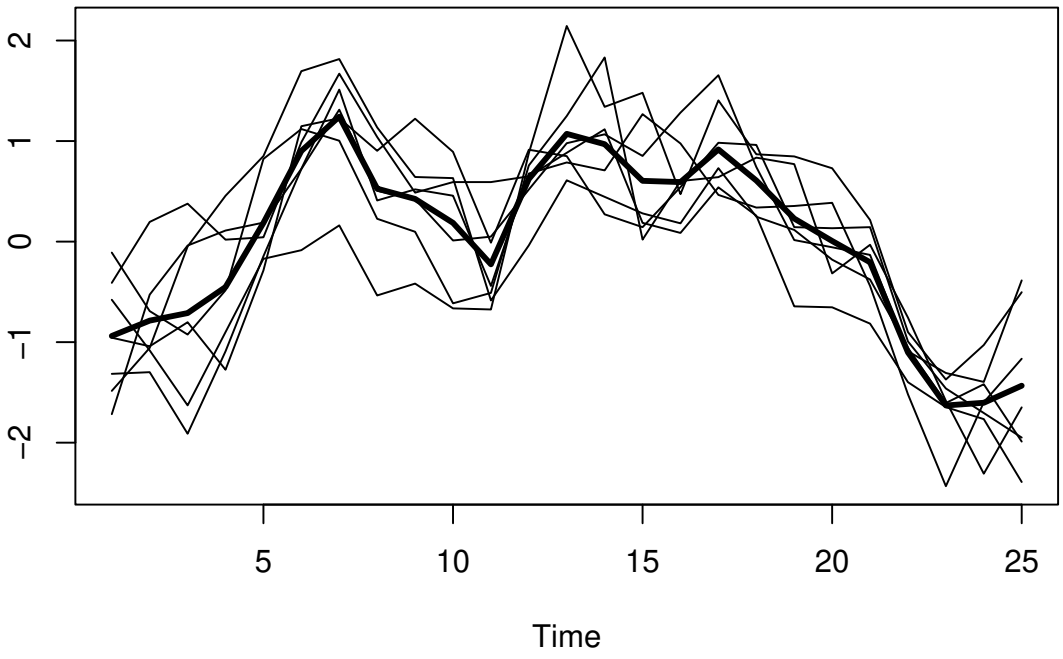
```
## Loading required package: dtw
## Loading required package: proxy
##
## Attaching package: 'proxy'
## The following objects are masked from 'package:stats':
##
##     as.dist, dist
## The following object is masked from 'package:base':
##
##     as.matrix
## Loaded dtw v1.23-1. See ?dtw for help, citation("dtw") for use in publication.
```

```
stocksDTW = dist(t(stockstsScaled), method = "DTW")
clusterTotalDTW = hclust(stocksDTW)
plot(clusterTotalDTW)
```

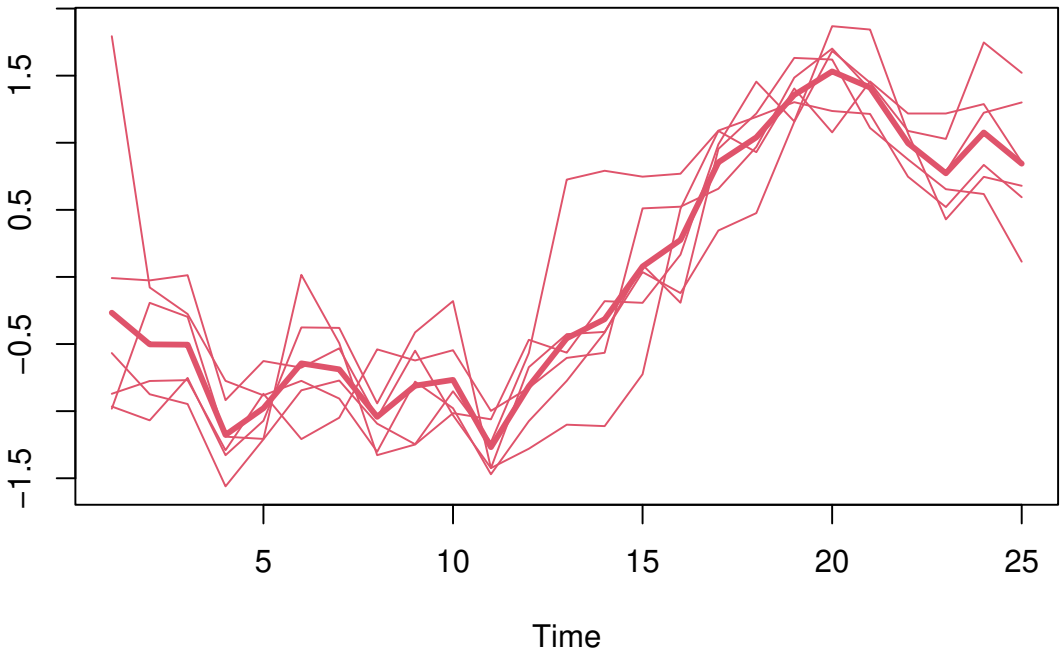


En este caso podríamos quedarnos con 3 o 5 grupos.

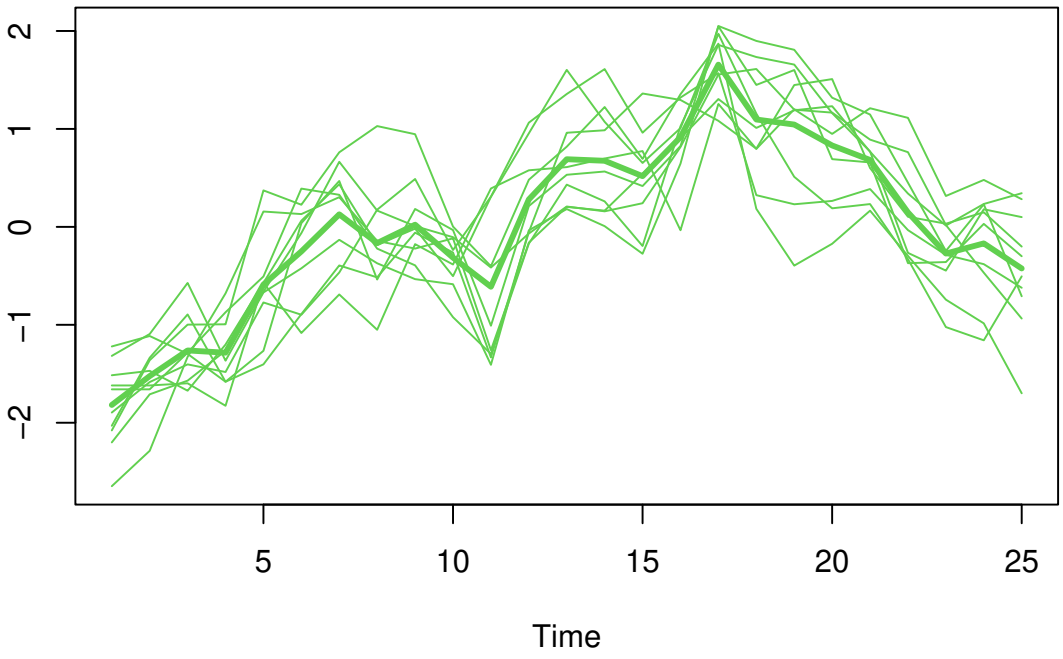
```
zDTW = cutree(clusterTotalDTW, 5)
m1 = apply(stockstsScaled[, zDTW == 1], 1, mean)
m2 = apply(stockstsScaled[, zDTW == 2], 1, mean)
m3 = apply(stockstsScaled[, zDTW == 3], 1, mean)
m4 = apply(stockstsScaled[, zDTW == 4], 1, mean)
m5 = apply(stockstsScaled[, zDTW == 5], 1, mean)
# A la vista del dendograma intuimos 3 grupos
require(useful)
ts.plot(cbind(m1, stockstsScaled[, zDTW == 1]), lwd = c(3, rep(1,
    sum(zDTW == 1))), col = 1)
```



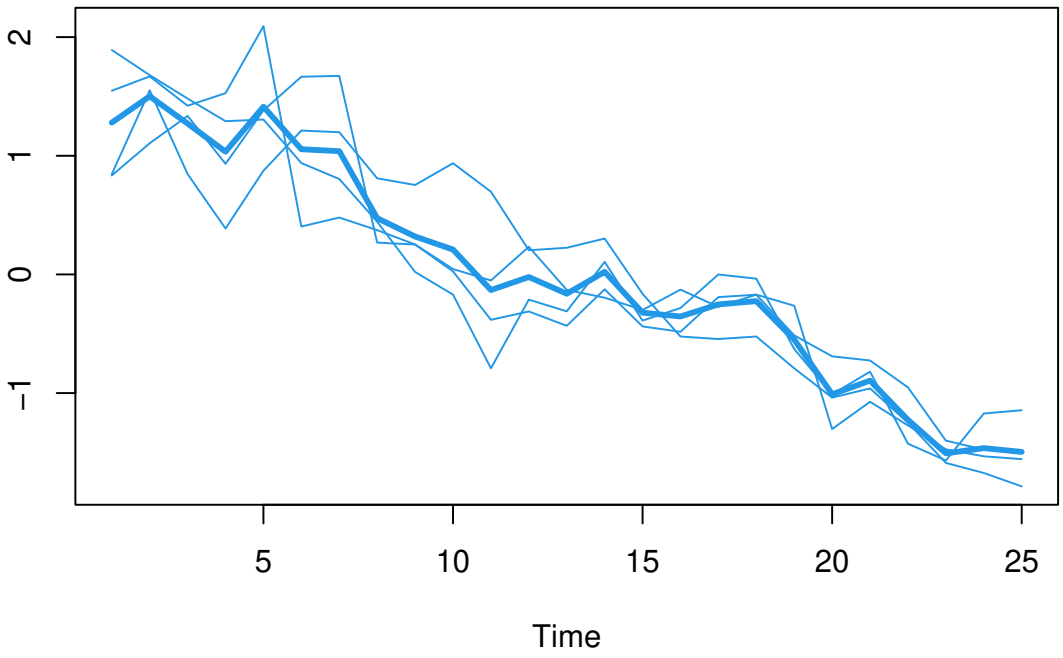

```
ts.plot(cbind(m2, stockstsScaled[, zDTW == 2]), lwd = c(3, rep(1,
    sum(zDTW == 2))), col = 2)
```



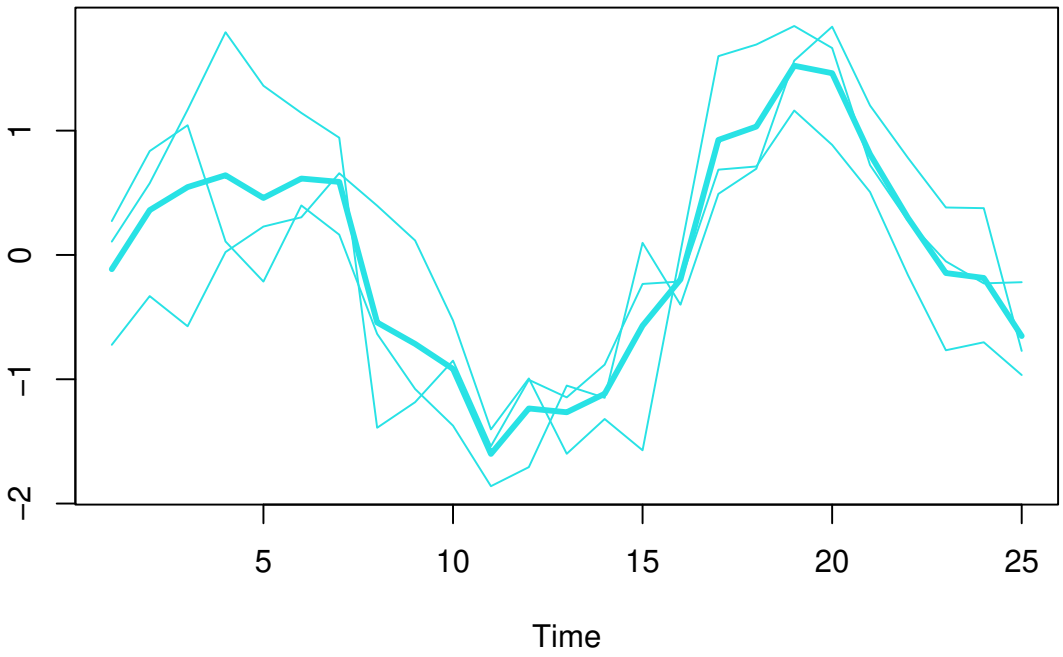

```
ts.plot(cbind(m3, stockstsScaled[, zDTW == 3]), lwd = c(3, rep(1,
    sum(zDTW == 3))), col = 3)
```




```
ts.plot(cbind(m4, stockstsScaled[, zDTW == 4]), lwd = c(3, rep(1,
    sum(zDTW == 4))), col = 4)
```




```
ts.plot(cbind(m5, stockstsScaled[, zDTW == 5]), lwd = c(3, rep(1,
    sum(zDTW == 5))), col = 5)
```



10. Identificar las diferencias entre los dos analisis

Pueden verse resultados similares.

Si bien el cluster 5, el nuevo, corresponde con valores que podrían haber sido considerados atípicos en el análisis anterior, correspondientes a las series: *INTC*, *PG* yy *WMT*.

Aprendizaje Automático I
Grado en Ciencia e Ingeniería de Datos
Universidad Rey Juan Carlos

Ejercicio: Medidas de rendimiento

DSLab

diciembre, 2023

El objetivo de este ejercicio es construir un **árbol de decisión** para el conjunto de datos `adult` de la librería `liver`.

1. Como primera parte del ejercicio, explora las características básicas de las variables en la base de datos.
2. Divide la base de datos en `entrenamiento` y `validación`. ¿Qué porcentaje eliges para cada partición? Justifícalo.
3. Empleando la función `trainControl` de la librería `caret` fija el número de particiones que consideres oportunas para entrenar el modelo en la de entrenamiento.
4. Entrena un modelo *knn* empleando como variables explicativas la edad (`age`) y el número de horas por semana `hours.per.week`. Evalua el rendimiento del modelo.
5. A continuación entrena un **árbol de decisión** empleando las variables que consideres oportunas. Justifica tu respuesta.
6. Obten una visualización del mejor árbol entrenado.
7. ¿Cómo funciona sobre la muestra de validación?

December 15, 2023

Solución ejercicio: Medidas de rendimiento

Los siguientes resultados han sido obtenidos con un script de R.

```
# Liberías necesarias para resolver el ejercicio
library(liver)

##
## Attaching package:  'liver'
## The following object is masked from 'package:base':
##
##     transform

library(caret)
library(caTools)
library(rpart.plot)

# Datos
data(adult)

# Resumen
summary(adult)

##       age              workclass      demogweight             education
##  Min.   :17.0   ?           : 2794   Min.   :  12285   HS-grad     :15750
##  1st Qu.:28.0   Gov         : 6536   1st Qu.: 117550   Some-college:10860
##  Median :37.0   Never-worked:   10   Median : 178215   Bachelors   : 7962
##  Mean   :38.6   Private     :33780   Mean   : 189685   Masters     : 2627
##  3rd Qu.:48.0   Self-emp    : 5457   3rd Qu.: 237713   Assoc-voc   : 2058
##  Max.   :90.0   Without-pay :   21   Max.   :1490400   11th        : 1812
##                                                       (Other)     : 7529
##  education.num        marital.status           occupation           relationship
##  Min.   : 1.00   Divorced     : 6613   Craft-repair   : 6096   Husband       :19537
##  1st Qu.: 9.00   Married      :22847   Prof-specialty : 6071   Not-in-family :12546
##  Median :10.00   Never-married:16096   Exec-managerial: 6019   Other-relative: 1506
##  Mean   :10.06   Separated    : 1526   Adm-clerical   : 5603   Own-child     : 7577
##  3rd Qu.:12.00   Widowed      : 1516   Sales          : 5470   Unmarried     : 5118
##  Max.   :16.00                         Other-service  : 4920   Wife          : 2314
##                                        (Other)        :14419
##                  race          gender       capital.gain      capital.loss
##  Amer-Indian-Eskimo:  470   Female:16156   Min.   :    0.0   Min.   :   0.00
##  Asian-Pac-Islander: 1504   Male  :32442   1st Qu.:    0.0   1st Qu.:   0.00
##  Black             : 4675                  Median :    0.0   Median :   0.00
##  Other             :  403                  Mean   :  582.4   Mean   :  87.94
##  White             :41546                  3rd Qu.:    0.0   3rd Qu.:   0.00
##                                            Max.   :41310.0   Max.   :4356.00
```

```
##
##  hours.per.week        native.country    income
##  Min.   : 1.00   United-States:43613   <=50K:37155
##  1st Qu.:40.00   Mexico       :  949   >50K :11443
##  Median :40.00   ?            :  847
##  Mean   :40.37   Philippines  :  292
##  3rd Qu.:45.00   Germany      :  206
##  Max.   :99.00   Puerto-Rico  :  184
##                  (Other)      : 2507
```

```
# Partición de los datos

# Mediante una semilla conseguimos que el ejercicio sea reproducible
set.seed(12321)

# Usamos el 20% de la base de datos como conjunto de entrenamiento y el resto como conjunto de validaci
sample = sample.split(adult$income, SplitRatio=0.2)
datos.train  = subset(adult, sample==TRUE)
datos.valid   = subset(adult, sample==FALSE)

# Entrenamos un modelo sobre la muestra de entrenamiento empleando todas las variables

traindata = datos.train[,-15]
trainclasses = datos.train[,15]
validdata = datos.valid[,-15]
validclasses = datos.valid[,15]

ctrl <- trainControl(method = "cv", number = 5)

# Entrenamos un knn

# Entrenamos un knn en cada una de las particiones
ctrl <- trainControl(method = "cv", number = 5)
traindata1 = as.data.frame(cbind(traindata$age,traindata$hours.per.week))
knn.fit1 = train(traindata1,trainclasses,method="knn",trControl=ctrl, preProcess = c("center","scale"))
knn.fit1
```

```
## k-Nearest Neighbors
##
## 9720 samples
##    2 predictor
##    2 classes: '<=50K', '>50K'
##
## Pre-processing: centered (2), scaled (2)
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 7776, 7776, 7775, 7777, 7776
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
##   5  0.7562751  0.1631052
##   7  0.7609056  0.1704753
##   9  0.7644034  0.1784621
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 9.
```

```r
# Modelo Final
knn.fit1$finalModel
```

```
## 9-nearest neighbor model
## Training set outcome distribution:
##
## <=50K  >50K
##  7431  2289
```

```r
# Resultados del modelo para cada una de las submuestras
knn.fit1$resample
```

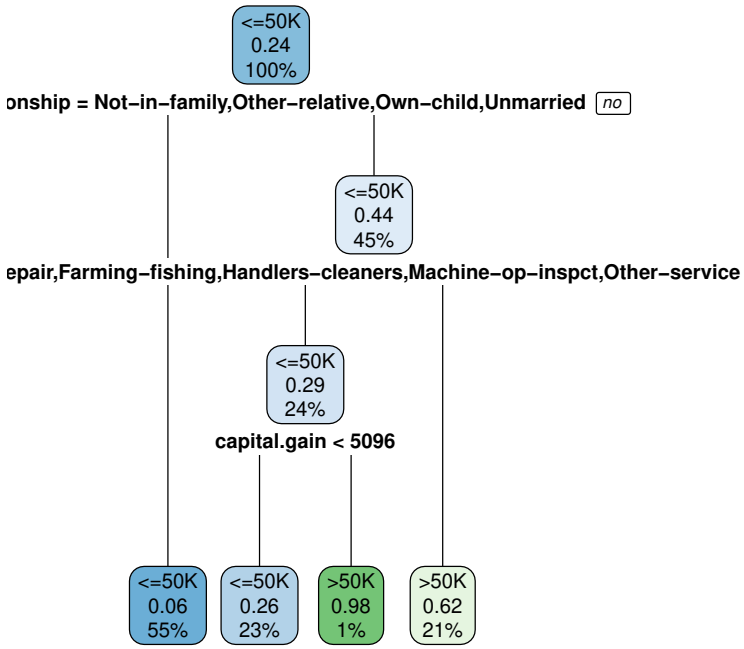
```
##    Accuracy     Kappa Resample
## 1 0.7664609 0.2006926    Fold1
## 2 0.7637674 0.1657707    Fold4
## 3 0.7629820 0.1784940    Fold3
## 4 0.7613169 0.1660047    Fold2
## 5 0.7674897 0.1813486    Fold5
```

```r
# Error de clasificación en train
# sobre la partición de entrenamiento
prediction = predict(knn.fit1$finalModel, traindata1, type = 'class')
cf = confusionMatrix(prediction, as.factor(trainclasses),positive=">50K")
print(cf)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction <=50K >50K
##      <=50K  7412 2287
##      >50K     19    2
##
##                Accuracy : 0.7628
##                  95% CI : (0.7542, 0.7712)
##     No Information Rate : 0.7645
##     P-Value [Acc > NIR] : 0.6628
##
##                   Kappa : -0.0026
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.0008737
##             Specificity : 0.9974431
##          Pos Pred Value : 0.0952381
##          Neg Pred Value : 0.7642025
##              Prevalence : 0.2354938
##          Detection Rate : 0.0002058
##    Detection Prevalence : 0.0021605
##       Balanced Accuracy : 0.4991584
##
##        'Positive' Class : >50K
##
```

```r
# Entrenamos un árbol en cada una de las particiones
dt.fit1 = train(traindata,trainclasses,method="rpart",trControl=ctrl)
dt.fit1
```

```
## CART
##
## 9720 samples
##   14 predictor
##    2 classes: '<=50K', '>50K'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 7775, 7776, 7777, 7776, 7776
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.03363914  0.8299390  0.4903102
##   0.04062910  0.8180055  0.4657887
##   0.10943644  0.7926955  0.2528124
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.03363914.

# Modelo Final
dt.fit1$finalModel

## n= 9720
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 9720 2289 <=50K (0.76450617 0.23549383)
##    2) relationship=Not-in-family,Other-relative,Own-child,Unmarried 5315  335 <=50K (0.93697084 0.063
##    3) relationship=Husband,Wife 4405 1954 <=50K (0.55641317 0.44358683)
##      6) occupation=?,Adm-clerical,Craft-repair,Farming-fishing,Handlers-cleaners,Machine-op-inspct,Ot
##       12) capital.gain< 5095.5 2239  574 <=50K (0.74363555 0.25636445) *
##       13) capital.gain>=5095.5 97    2 >50K (0.02061856 0.97938144) *
##      7) occupation=Armed-Forces,Exec-managerial,Prof-specialty,Protective-serv,Sales,Tech-support 206

rpart.plot(dt.fit1$finalModel)
```




```
# Resultados del modelo para cada una de las submuestras
dt.fit1$resample
```

```
##    Accuracy     Kappa Resample
## 1 0.8313625 0.4635475    Fold1
## 2 0.8179012 0.4393843    Fold2
## 3 0.8353909 0.4999293    Fold5
## 4 0.8266461 0.5040981    Fold4
## 5 0.8383942 0.5445919    Fold3
```

```
# Error de clasificación en train
# sobre la partición de entrenamiento
prediction = predict(dt.fit1$finalModel, datos.train, type = 'class')
cf = confusionMatrix(prediction, as.factor(trainclasses),positive=">50K")
print(cf)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction <=50K >50K
##      <=50K  6645  909
##      >50K    786 1380
##
##                Accuracy : 0.8256
##                  95% CI : (0.8179, 0.8331)
##     No Information Rate : 0.7645
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5065
##
```

```
##  Mcnemar's Test P-Value : 0.003044
##
##             Sensitivity : 0.6029
##             Specificity : 0.8942
##          Pos Pred Value : 0.6371
##          Neg Pred Value : 0.8797
##              Prevalence : 0.2355
##          Detection Rate : 0.1420
##    Detection Prevalence : 0.2228
##       Balanced Accuracy : 0.7486
##
##        'Positive' Class : >50K
##
```

```
# sobre la partición de validación
prediction = predict(dt.fit1$finalModel, datos.valid, type = 'class')
cf = confusionMatrix(prediction, as.factor(validclasses),positive=">50K")
print(cf)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction <=50K  >50K
##      <=50K 26700  3662
##      >50K   3024  5492
##
##                Accuracy : 0.828
##                  95% CI : (0.8242, 0.8318)
##     No Information Rate : 0.7645
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5105
##
##  Mcnemar's Test P-Value : 6.683e-15
##
##             Sensitivity : 0.6000
##             Specificity : 0.8983
##          Pos Pred Value : 0.6449
##          Neg Pred Value : 0.8794
##              Prevalence : 0.2355
##          Detection Rate : 0.1413
##    Detection Prevalence : 0.2190
##       Balanced Accuracy : 0.7491
##
##        'Positive' Class : >50K
##
```

Información de la sesión de R (incluyendo información sobre el sistema operativo, la versión de R y los paquetes usados):

```
sessionInfo()
```

```
## R version 4.3.1 (2023-06-16)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 20.04.6 LTS
```

```
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/atlas/libblas.so.3.10.3
## LAPACK: /usr/lib/x86_64-linux-gnu/atlas/liblapack.so.3.10.3;  LAPACK version 3.9.0
##
## locale:
##  [1] LC_CTYPE=es_ES.UTF-8       LC_NUMERIC=C               LC_TIME=es_ES.UTF-8
##  [4] LC_COLLATE=es_ES.UTF-8     LC_MONETARY=es_ES.UTF-8    LC_MESSAGES=es_ES.UTF-8
##  [7] LC_PAPER=es_ES.UTF-8       LC_NAME=C                  LC_ADDRESS=C
## [10] LC_TELEPHONE=C             LC_MEASUREMENT=es_ES.UTF-8 LC_IDENTIFICATION=C
##
## time zone: Europe/Madrid
## tzcode source: system (glibc)
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] liver_1.15       ggfortify_0.4.16 factoextra_1.0.7 mlbench_2.1-3.1  readxl_1.4.3
##  [6] caret_6.0-94     lattice_0.21-9   ggplot2_3.4.3    rpart.plot_3.1.1 rpart_4.1.19
## [11] caTools_1.18.2   dplyr_1.1.3      ISLR2_1.3-2
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.2.0     timeDate_4022.108    farver_2.1.1         bitops_1.0-7
##  [5] fastmap_1.1.1        pROC_1.18.4          digest_0.6.33        timechange_0.2.0
##  [9] lifecycle_1.0.3      survival_3.5-7       magrittr_2.0.3       compiler_4.3.1
## [13] rlang_1.1.1          tools_4.3.1          utf8_1.2.3           yaml_2.3.7
## [17] data.table_1.14.8    knitr_1.44           labeling_0.4.3       plyr_1.8.9
## [21] withr_2.5.1          purrr_1.0.2          nnet_7.3-19          grid_4.3.1
## [25] stats4_4.3.1         fansi_1.0.5          e1071_1.7-13         colorspace_2.1-0
## [29] future_1.33.0        globals_0.16.2       scales_1.2.1         iterators_1.0.14
## [33] MASS_7.3-60          tinytex_0.47         cli_3.6.1            rmarkdown_2.25
## [37] generics_0.1.3       rstudioapi_0.15.0    future.apply_1.11.0  reshape2_1.4.4
## [41] tzdb_0.4.0           proxy_0.4-27         stringr_1.5.0        splines_4.3.1
## [45] parallel_4.3.1       cellranger_1.1.0     vctrs_0.6.3          hardhat_1.3.0
## [49] Matrix_1.6-1.1       hms_1.1.3            ggrepel_0.9.3        listenv_0.9.0
## [53] foreach_1.5.2        tidyr_1.3.0          gower_1.0.1          recipes_1.0.8
## [57] glue_1.6.2           parallelly_1.36.0    codetools_0.2-19     lubridate_1.9.3
## [61] stringi_1.7.12       gtable_0.3.4         munsell_0.5.0        tibble_3.2.1
## [65] pillar_1.9.0         htmltools_0.5.6.1    ipred_0.9-14         lava_1.7.2.1
## [69] R6_2.5.1             evaluate_0.22        readr_2.1.4          highr_0.10
## [73] class_7.3-22         Rcpp_1.0.11          gridExtra_2.3        nlme_3.1-163
## [77] prodlim_2023.08.28   xfun_0.40            pkgconfig_2.0.3      ModelMetrics_1.2.2.2
```

```
Sys.time()
```

```
## [1] "2023-11-02 17:28:54 CET"
```

DSLAB
DATA SCIENCE LABORATORY

Aprendizaje Automático I
Grado en Ciencia e Ingeniería de Datos
Universidad Rey Juan Carlos

Ejercicio: Árboles de Decisión

DSLab

diciembre, 2023

El objetivo de este ejercicio es construir un **árbol de decisión** para el conjunto de datos `Carseats`de la libreria `ISLR2`.

1. Como primera parte del ejercicio, explora las características básicas de las variables en la base de datos. Además, construye una nueva variable respuesta `High`de tal forma que:

$$High = \begin{cases} "No", & \text{si Sales} \leq 8 \\ "Yes", & \text{otro caso} \end{cases}$$

2. Divide la base de datos en `entrenamiento` y `validación`. ¿Qué porcentaje eliges para cada partición? Justifícalo.
3. A continuación entrena un **árbol de decisión** empleando las variables que consideres oportunas. Justifica tu respuesta.
4. Obten una visualización del árbol entrenado.
5. ¿Qué variable es la más importante para conseguir ventas elevadas? ¿Qué valores de dicha variable están asociados a un mayor número de ventas? Obten una visualización que refleje ese hecho.
6. ¿Cuál es el error de clasificación? ¿Qué valor es mayor, la **precisión** o la **recuperación**? ¿Qué significado tiene?

7. Si existe sobreajuste en el modelo, ¿qué podrías hacer para corregirlo?

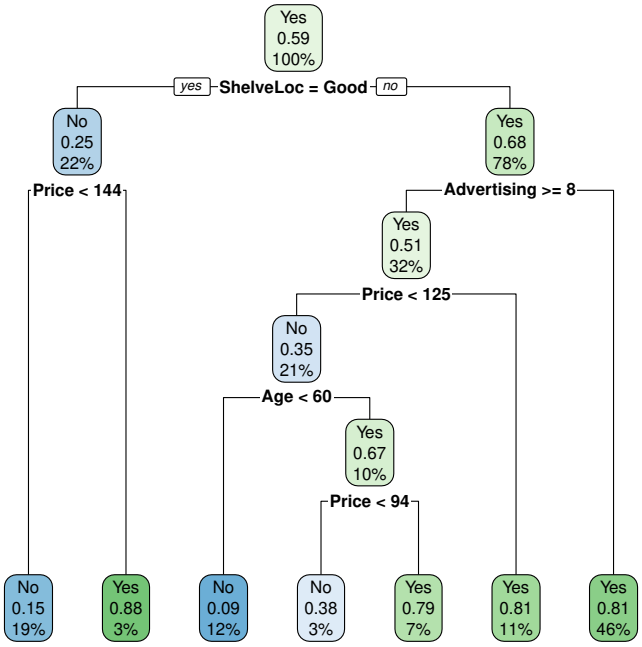
December 15, 2023

Solución ejercicio: Árboles de Decisión

Los siguientes resultados han sido obtenidos con un script de R.

```
# Liberías necesarias para resolver el ejercicio
library(ISLR2)
library(dplyr)
library(caTools) # Particiones de los datos
library(rpart) # Para árboles de decisión
library(rpart.plot)
library(ggplot2)
library(caret) # Para la matriz de confusión


# Datos
attach(Carseats)

## The following objects are masked from Carseats (pos = 3):
##
##     Advertising, Age, CompPrice, Education, Income, Population, Price, Sales,
##     ShelveLoc, Urban, US
## The following objects are masked from Carseats (pos = 4):
##
##     Advertising, Age, CompPrice, Education, Income, Population, Price, Sales,
##     ShelveLoc, Urban, US
## The following objects are masked from Carseats (pos = 8):
##
##     Advertising, Age, CompPrice, Education, Income, Population, Price, Sales,
##     ShelveLoc, Urban, US

# Creamos una nueva variable respuesta binaria


# Creamos el data frame

df = Carseats %>%
      mutate(High = factor(ifelse(Sales>=8,"No","Yes"))) %>%
      select(-Sales)

# Partición de los datos

# Mediante una semilla conseguimos que el ejercicio sea reproducible
set.seed(121)

# Usamos el 70% de la base de datos como conjunto de entrenamiento y el resto como conjunto de test
```

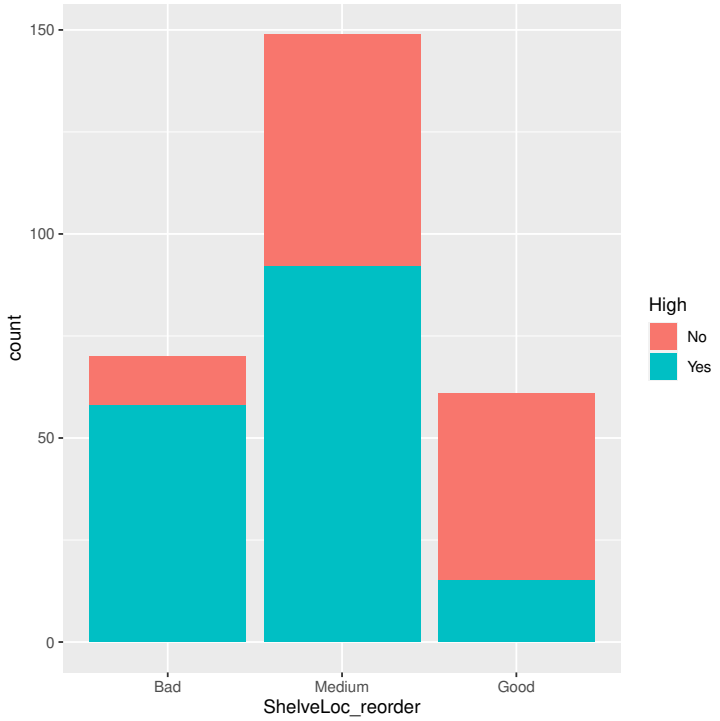
```r
sample = sample.split(df$High, SplitRatio=0.7)
train  = subset(df, sample==TRUE)
test   = subset(df, sample==FALSE)

# Entrenamos un modelo sobre la muestra de entrenamiento empleando todas las variables

fit.dt = rpart(High~., data = train, method = 'class')
rpart.plot(fit.dt, extra = 106)
```



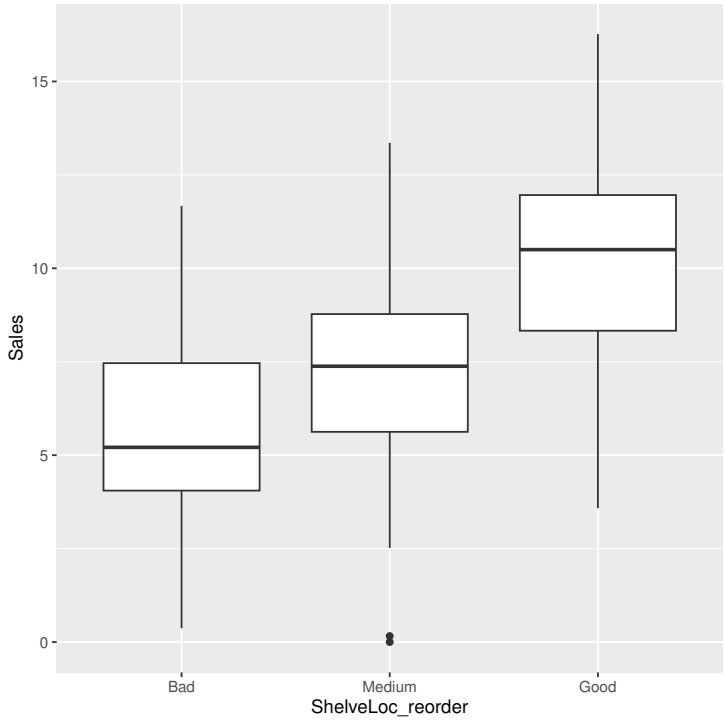

```r
# La variable más importante es:
fit.dt$variable.importance
```

```
##   ShelveLoc       Price Advertising         Age   CompPrice          US   Education
##   19.486155   18.342365   10.575532    9.955631    6.354683    4.737580    2.558957
##  Population      Income
##    2.547655    1.973958
```

```r
# Relación entre la variable respuesta y la variable más importante
# reordenamos la variable
train %>%
  mutate(ShelveLoc_reorder=factor(ShelveLoc,levels=c("Bad","Medium","Good")))%>%
  ggplot(aes(x = ShelveLoc_reorder, fill = High)) +
  geom_bar()
```




```
# Podemos visualizar su relación con la variable respuesta original como sigue
# reordenamos la variable
df %>%
  mutate(ShelveLoc_reorder=factor(ShelveLoc,levels=c("Bad","Medium","Good")))%>%
  ggplot(aes(ShelveLoc_reorder, Sales)) +
  geom_boxplot()
```




```
# Error de clasificación en train
# sobre la partición de entrenamiento
prediction = predict(fit.dt, train, type = 'class')
cf = confusionMatrix(prediction, as.factor(train$High),positive="Yes")
print(cf)
```

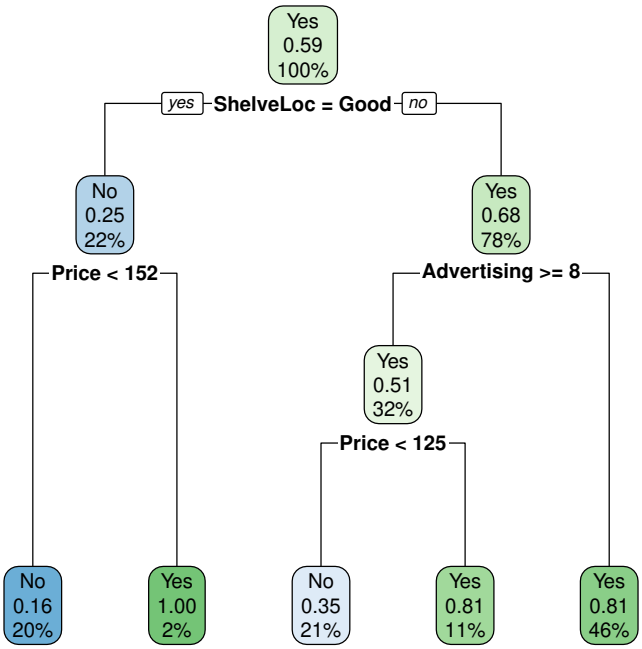
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No   80  14
##        Yes  35 151
##
##                Accuracy : 0.825
##                  95% CI : (0.7753, 0.8676)
##     No Information Rate : 0.5893
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6282
##
##  Mcnemar's Test P-Value : 0.004275
##
##             Sensitivity : 0.9152
##             Specificity : 0.6957
##          Pos Pred Value : 0.8118
##          Neg Pred Value : 0.8511
##              Prevalence : 0.5893
##          Detection Rate : 0.5393
##    Detection Prevalence : 0.6643
##       Balanced Accuracy : 0.8054
```

```
##
##        'Positive' Class : Yes
##
```

```
# sobre la partición de validación
prediction = predict(fit.dt, test, type = 'class')
cf = confusionMatrix(prediction, as.factor(test$High),positive="Yes")
print(cf)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction No Yes
##        No  27  10
##        Yes 22  61
##
##                Accuracy : 0.7333
##                  95% CI : (0.6449, 0.8099)
##     No Information Rate : 0.5917
##     P-Value [Acc > NIR] : 0.0008589
##
##                   Kappa : 0.4264
##
##  Mcnemar's Test P-Value : 0.0518299
##
##             Sensitivity : 0.8592
##             Specificity : 0.5510
##          Pos Pred Value : 0.7349
##          Neg Pred Value : 0.7297
##              Prevalence : 0.5917
##          Detection Rate : 0.5083
##    Detection Prevalence : 0.6917
##       Balanced Accuracy : 0.7051
##
##        'Positive' Class : Yes
##
```

```
# Ajustamos un modelo con menos profundidad para evitar el sobreajuste.
control = rpart.control(minsplit = 4,
                        minbucket = round(5 / 3),
                        maxdepth = 3,
                        cp = 0)
tune.fit = rpart(High~., data = train, method = 'class', control = control)
rpart.plot(tune.fit, extra = 106)
```




```
# Error de clasificación en train
# sobre la partición de entrenamiento
prediction = predict(tune.fit, train, type = 'class')
cf = confusionMatrix(prediction, as.factor(train$High),positive="Yes")
print(cf)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No   85  30
##        Yes  30 135
##
##                Accuracy : 0.7857
##                  95% CI : (0.733, 0.8323)
##     No Information Rate : 0.5893
##     P-Value [Acc > NIR] : 2.758e-12
##
##                   Kappa : 0.5573
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.8182
##             Specificity : 0.7391
##          Pos Pred Value : 0.8182
##          Neg Pred Value : 0.7391
##              Prevalence : 0.5893
##          Detection Rate : 0.4821
##    Detection Prevalence : 0.5893
##       Balanced Accuracy : 0.7787
```

```
##
##        'Positive' Class : Yes
##

# sobre la partición de validación
prediction = predict(tune.fit, test, type = 'class')
cf = confusionMatrix(prediction, as.factor(test$High),positive="Yes")
print(cf)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction No Yes
##        No  32  15
##        Yes 17  56
##
##                Accuracy : 0.7333
##                  95% CI : (0.6449, 0.8099)
##     No Information Rate : 0.5917
##     P-Value [Acc > NIR] : 0.0008589
##
##                   Kappa : 0.4446
##
##  Mcnemar's Test P-Value : 0.8596838
##
##             Sensitivity : 0.7887
##             Specificity : 0.6531
##          Pos Pred Value : 0.7671
##          Neg Pred Value : 0.6809
##              Prevalence : 0.5917
##          Detection Rate : 0.4667
##    Detection Prevalence : 0.6083
##       Balanced Accuracy : 0.7209
##
##        'Positive' Class : Yes
##
```

Información de la sesión de R (incluyendo información sobre el sistema operativo, la versión de R y los paquetes usados):

```
sessionInfo()

## R version 4.3.1 (2023-06-16)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 20.04.6 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/atlas/libblas.so.3.10.3
## LAPACK: /usr/lib/x86_64-linux-gnu/atlas/liblapack.so.3.10.3;  LAPACK version 3.9.0
##
## locale:
##  [1] LC_CTYPE=es_ES.UTF-8       LC_NUMERIC=C               LC_TIME=es_ES.UTF-8
##  [4] LC_COLLATE=es_ES.UTF-8     LC_MONETARY=es_ES.UTF-8    LC_MESSAGES=es_ES.UTF-8
##  [7] LC_PAPER=es_ES.UTF-8       LC_NAME=C                  LC_ADDRESS=C
## [10] LC_TELEPHONE=C             LC_MEASUREMENT=es_ES.UTF-8 LC_IDENTIFICATION=C
```

```
##
## time zone: Europe/Madrid
## tzcode source: system (glibc)
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] caret_6.0-94     lattice_0.21-9   ggplot2_3.4.3    rpart.plot_3.1.1 rpart_4.1.19
## [6] caTools_1.18.2   dplyr_1.1.3      ISLR2_1.3-2
##
## loaded via a namespace (and not attached):
##  [1] gtable_0.3.4         xfun_0.40            recipes_1.0.8        tzdb_0.4.0
##  [5] vctrs_0.6.3          tools_4.3.1          bitops_1.0-7         generics_0.1.3
##  [9] stats4_4.3.1         parallel_4.3.1       proxy_0.4-27         tibble_3.2.1
## [13] fansi_1.0.5          highr_0.10           ModelMetrics_1.2.2.2 pkgconfig_2.0.3
## [17] Matrix_1.6-1.1       data.table_1.14.8    lifecycle_1.0.3      stringr_1.5.0
## [21] compiler_4.3.1       farver_2.1.1         tinytex_0.47         munsell_0.5.0
## [25] codetools_0.2-19     htmltools_0.5.6.1    class_7.3-22         yaml_2.3.7
## [29] prodlim_2023.08.28   pillar_1.9.0         MASS_7.3-60          gower_1.0.1
## [33] iterators_1.0.14     foreach_1.5.2        nlme_3.1-163         parallelly_1.36.0
## [37] lava_1.7.2.1         tidyselect_1.2.0     digest_0.6.33        stringi_1.7.12
## [41] future_1.33.0        reshape2_1.4.4       purrr_1.0.2          listenv_0.9.0
## [45] labeling_0.4.3       splines_4.3.1        fastmap_1.1.1        grid_4.3.1
## [49] colorspace_2.1-0     cli_3.6.1            magrittr_2.0.3       survival_3.5-7
## [53] utf8_1.2.3           e1071_1.7-13         future.apply_1.11.0  readr_2.1.4
## [57] withr_2.5.1          scales_1.2.1         lubridate_1.9.3      timechange_0.2.0
## [61] rmarkdown_2.25       globals_0.16.2       nnet_7.3-19          timeDate_4022.108
## [65] hms_1.1.3            evaluate_0.22        knitr_1.44           hardhat_1.3.0
## [69] rlang_1.1.1          Rcpp_1.0.11          glue_1.6.2           pROC_1.18.4
## [73] ipred_0.9-14         rstudioapi_0.15.0    R6_2.5.1             plyr_1.8.9
```

```
Sys.time()
```

```
## [1] "2023-11-01 19:20:14 CET"
```

Aprendizaje Automático I
Grado en Ciencia e Ingeniería de Datos
Universidad Rey Juan Carlos

Ejercicio: Regresión Logística

DSLAB

diciembre, 2023

Estos datos abordan el rendimiento de los alumnos de enseñanza secundaria de dos centros portugueses. Podéis descargar los datos aquí: **Student Data**. Las características de los datos incluyen las calificaciones de los alumnos, características demográficas, sociales y relacionadas con el centro escolar) y se recogieron mediante informes escolares y cuestionarios. Se proporcionan dos conjuntos de datos relativos al rendimiento en dos asignaturas distintas: Matemáticas (mat) y Lengua portuguesa (por).

1. Leer los datos y unirlos en un único `data frame`.

```
df1=read.table("./datos/student-mat.csv",sep=";",header=TRUE)
df2=read.table("./datos/student-por.csv",sep=";",header=TRUE)

df3=merge(df1,df2,by=c("school","sex","age","address","famsize",
                       "Pstatus","Medu","Fedu","Mjob","Fjob","reason",
                       "nursery","internet"))
print(nrow(df3)) # 382 students
```

2. Análisis exploratorio de los datos. Visualiza la variable respuesta `G3.y`. Un valor de 0 indica que el alumno ha abandonado la asigantura. Estas observaciones han de ser elminadas.

3. Construye una nueva variable.

$$final = \begin{cases} "pass", & \text{si G3.y} \geq 10 \\ "fail", & \text{otro caso} \end{cases}$$

4. Divide la base de datos en train (60%) y test (40%).
5. Estudia la relación de la variable `final` con la variable `Fedu`. ¿Tiene sentido unir agrupar alguna de las características de la variable? Discute la misma cuestión cuando se plantea sobre la variable `Medu`.
6. Estudia mediante un modelo de regresión logística cómo se modifican la relación entre `Fedu` y `Medu` con la variable respuesta cuando se considera una relación multivariante.
7. Aplica un proceso de selección de variables mediante la función `step`. ¿Qué variables tiene el modelo final? Entrena el modelo empleando k-fold.
8. Evalua el rendimiento del modelo.

December 15, 2023

Solución ejercicio: Regresión Logística

Los siguientes resultados han sido obtenidos con un script de R.

```
library(ggplot2)

df1=read.table("./datos/student-mat.csv",sep=";",header=TRUE)
df2=read.table("./datos/student-por.csv",sep=";",header=TRUE)

df3=merge(df1,df2,by=c("school","sex","age","address","famsize","Pstatus","Medu","Fedu","Mjob","Fjob","r
print(nrow(df3)) # 382 students

## [1] 382

# Limpieza de datos
# Resumen de datos
summary(df3)

##     school              sex                 age          address
##  Length:382         Length:382         Min.   :15.00   Length:382
##  Class :character   Class :character   1st Qu.:16.00   Class :character
##  Mode  :character   Mode  :character   Median :17.00   Mode  :character
##                                        Mean   :16.59
##                                        3rd Qu.:17.00
##                                        Max.   :22.00
##    famsize            Pstatus              Medu            Fedu           Mjob
##  Length:382         Length:382         Min.   :0.000   Min.   :0.000   Length:382
##  Class :character   Class :character   1st Qu.:2.000   1st Qu.:2.000   Class :character
##  Mode  :character   Mode  :character   Median :3.000   Median :3.000   Mode  :character
##                                        Mean   :2.806   Mean   :2.565
##                                        3rd Qu.:4.000   3rd Qu.:4.000
##                                        Max.   :4.000   Max.   :4.000
##      Fjob              reason            nursery            internet
##  Length:382         Length:382         Length:382         Length:382
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##   guardian.x         traveltime.x    studytime.x      failures.x     schoolsup.x
##  Length:382         Min.   :1.000   Min.   :1.000   Min.   :0.0000   Length:382
##  Class :character   1st Qu.:1.000   1st Qu.:1.000   1st Qu.:0.0000   Class :character
##  Mode  :character   Median :1.000   Median :2.000   Median :0.0000   Mode  :character
##                     Mean   :1.442   Mean   :2.034   Mean   :0.2906
##                     3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.:0.0000
##                     Max.   :4.000   Max.   :4.000   Max.   :3.0000
```

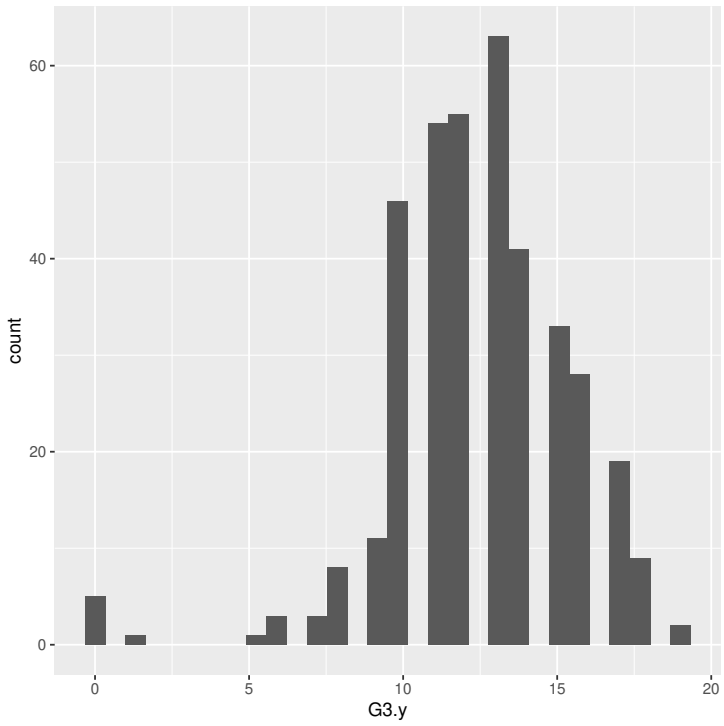
```
##    famsup.x            paid.x          activities.x         higher.x
##  Length:382         Length:382         Length:382         Length:382
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##   romantic.x           famrel.x      freetime.x       goout.x          Dalc.x
##  Length:382         Min.   :1.00   Min.   :1.000   Min.   :1.000   Min.   :1.000
##  Class :character   1st Qu.:4.00   1st Qu.:3.000   1st Qu.:2.000   1st Qu.:1.000
##  Mode  :character   Median :4.00   Median :3.000   Median :3.000   Median :1.000
##                     Mean   :3.94   Mean   :3.223   Mean   :3.113   Mean   :1.474
##                     3rd Qu.:5.00   3rd Qu.:4.000   3rd Qu.:4.000   3rd Qu.:2.000
##                     Max.   :5.00   Max.   :5.000   Max.   :5.000   Max.   :5.000
##      Walc.x        health.x       absences.x          G1.x            G2.x
##  Min.   :1.00   Min.   :1.000   Min.   : 0.000   Min.   : 3.00   Min.   : 0.00
##  1st Qu.:1.00   1st Qu.:3.000   1st Qu.: 0.000   1st Qu.: 8.00   1st Qu.: 8.25
##  Median :2.00   Median :4.000   Median : 3.000   Median :10.50   Median :11.00
##  Mean   :2.28   Mean   :3.579   Mean   : 5.319   Mean   :10.86   Mean   :10.71
##  3rd Qu.:3.00   3rd Qu.:5.000   3rd Qu.: 8.000   3rd Qu.:13.00   3rd Qu.:13.00
##  Max.   :5.00   Max.   :5.000   Max.   :75.000   Max.   :19.00   Max.   :19.00
##       G3.x        guardian.y         traveltime.y    studytime.y      failures.y
##  Min.   : 0.00   Length:382         Min.   :1.000   Min.   :1.000   Min.   :0.0000
##  1st Qu.: 8.00   Class :character   1st Qu.:1.000   1st Qu.:1.000   1st Qu.:0.0000
##  Median :11.00   Mode  :character   Median :1.000   Median :2.000   Median :0.0000
##  Mean   :10.39                      Mean   :1.445   Mean   :2.039   Mean   :0.1414
##  3rd Qu.:14.00                      3rd Qu.:2.000   3rd Qu.:2.000   3rd Qu.:0.0000
##  Max.   :20.00                      Max.   :4.000   Max.   :4.000   Max.   :3.0000
##  schoolsup.y          famsup.y           paid.y          activities.y
##  Length:382         Length:382         Length:382         Length:382
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##    higher.y          romantic.y           famrel.y       freetime.y      goout.y
##  Length:382         Length:382         Min.   :1.000   Min.   :1.00   Min.   :1.000
##  Class :character   Class :character   1st Qu.:4.000   1st Qu.:3.00   1st Qu.:2.000
##  Mode  :character   Mode  :character   Median :4.000   Median :3.00   Median :3.000
##                                        Mean   :3.942   Mean   :3.23   Mean   :3.118
##                                        3rd Qu.:5.000   3rd Qu.:4.00   3rd Qu.:4.000
##                                        Max.   :5.000   Max.   :5.00   Max.   :5.000
##      Dalc.y          Walc.y         health.y       absences.y          G1.y
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   : 0.000   Min.   : 0.00
##  1st Qu.:1.000   1st Qu.:1.000   1st Qu.:3.000   1st Qu.: 0.000   1st Qu.:10.00
##  Median :1.000   Median :2.000   Median :4.000   Median : 2.000   Median :12.00
##  Mean   :1.476   Mean   :2.291   Mean   :3.576   Mean   : 3.673   Mean   :12.11
##  3rd Qu.:2.000   3rd Qu.:3.000   3rd Qu.:5.000   3rd Qu.: 6.000   3rd Qu.:14.00
##  Max.   :5.000   Max.   :5.000   Max.   :5.000   Max.   :32.000   Max.   :19.00
##       G2.y            G3.y
##  Min.   : 5.00   Min.   : 0.00
##  1st Qu.:11.00   1st Qu.:11.00
##  Median :12.00   Median :13.00
##  Mean   :12.24   Mean   :12.52
```

```
##  3rd Qu.:14.00   3rd Qu.:14.00
##  Max.   :19.00   Max.   :19.00

str(df3)

## 'data.frame': 382 obs. of  53 variables:
##  $ school      : chr  "GP" "GP" "GP" "GP" ...
##  $ sex         : chr  "F" "F" "F" "F" ...
##  $ age         : int  15 15 15 15 15 15 15 15 15 15 ...
##  $ address     : chr  "R" "R" "R" "R" ...
##  $ famsize     : chr  "GT3" "GT3" "GT3" "GT3" ...
##  $ Pstatus     : chr  "T" "T" "T" "T" ...
##  $ Medu        : int  1 1 2 2 3 3 3 2 3 3 ...
##  $ Fedu        : int  1 1 2 4 3 4 4 2 1 3 ...
##  $ Mjob        : chr  "at_home" "other" "at_home" "services" ...
##  $ Fjob        : chr  "other" "other" "other" "health" ...
##  $ reason      : chr  "home" "reputation" "reputation" "course" ...
##  $ nursery     : chr  "yes" "no" "yes" "yes" ...
##  $ internet    : chr  "yes" "yes" "no" "yes" ...
##  $ guardian.x  : chr  "mother" "mother" "mother" "mother" ...
##  $ traveltime.x: int  2 1 1 1 2 1 2 2 2 1 ...
##  $ studytime.x : int  4 2 1 3 3 3 3 2 4 4 ...
##  $ failures.x  : int  1 2 0 0 2 0 2 0 0 0 ...
##  $ schoolsup.x : chr  "yes" "yes" "yes" "yes" ...
##  $ famsup.x    : chr  "yes" "yes" "yes" "yes" ...
##  $ paid.x      : chr  "yes" "no" "yes" "yes" ...
##  $ activities.x: chr  "yes" "no" "yes" "yes" ...
##  $ higher.x    : chr  "yes" "yes" "yes" "yes" ...
##  $ romantic.x  : chr  "no" "yes" "no" "no" ...
##  $ famrel.x    : int  3 3 4 4 4 4 4 4 4 4 ...
##  $ freetime.x  : int  1 3 3 3 2 3 2 1 4 3 ...
##  $ goout.x     : int  2 4 1 2 1 2 2 3 2 3 ...
##  $ Dalc.x      : int  1 2 1 1 2 1 2 1 2 1 ...
##  $ Walc.x      : int  1 4 1 1 3 1 2 3 3 1 ...
##  $ health.x    : int  1 5 2 5 3 5 5 4 3 4 ...
##  $ absences.x  : int  2 2 8 2 8 2 0 2 12 10 ...
##  $ G1.x        : int  7 8 14 10 10 12 12 8 16 10 ...
##  $ G2.x        : int  10 6 13 9 10 12 0 9 16 11 ...
##  $ G3.x        : int  10 5 13 8 10 11 0 8 16 11 ...
##  $ guardian.y  : chr  "mother" "mother" "mother" "mother" ...
##  $ traveltime.y: int  2 1 1 1 2 1 2 2 2 1 ...
##  $ studytime.y : int  4 2 1 3 3 3 3 2 4 4 ...
##  $ failures.y  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ schoolsup.y : chr  "yes" "yes" "yes" "yes" ...
##  $ famsup.y    : chr  "yes" "yes" "yes" "yes" ...
##  $ paid.y      : chr  "yes" "no" "no" "no" ...
##  $ activities.y: chr  "yes" "no" "yes" "yes" ...
##  $ higher.y    : chr  "yes" "yes" "yes" "yes" ...
##  $ romantic.y  : chr  "no" "yes" "no" "no" ...
##  $ famrel.y    : int  3 3 4 4 4 4 4 4 4 4 ...
##  $ freetime.y  : int  1 3 3 3 2 3 2 1 4 3 ...
##  $ goout.y     : int  2 4 1 2 1 2 2 3 2 3 ...
##  $ Dalc.y      : int  1 2 1 1 2 1 2 1 2 1 ...
##  $ Walc.y      : int  1 4 1 1 3 1 2 3 3 1 ...
```

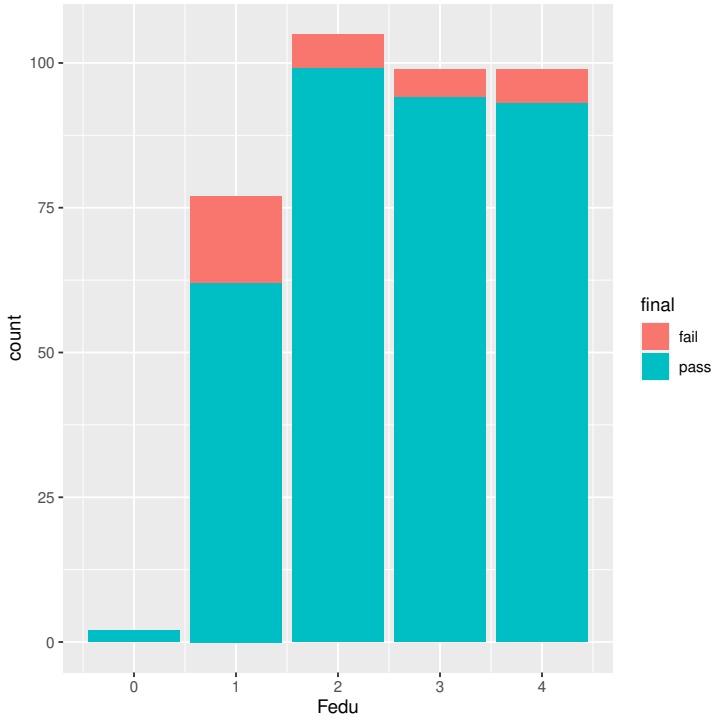
```
##  $ health.y    : int  1 5 2 5 3 5 5 4 3 4 ...
##  $ absences.y  : int  4 2 8 2 2 2 0 0 6 10 ...
##  $ G1.y        : int  13 13 14 10 13 11 10 11 15 10 ...
##  $ G2.y        : int  13 11 13 11 13 12 11 10 15 10 ...
##  $ G3.y        : int  13 11 12 10 13 12 12 11 15 10 ...
```

```
# Visualización de datos
df3 %>% ggplot() +
  geom_histogram(mapping = aes(x = G3.y))
```

```
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
```




```
# nueva variable respuesta
df3$final <- factor(ifelse(df3$G3.y >= 10, 1, 0), labels = c("fail", "pass"))

# Fedu
ggplot(df3, aes(x=Fedu, group=final,fill=final)) + geom_bar()
```




```
# Partición de datos
# mediante una semilla conseguimos que el ejercicio sea reproducible
set.seed(1)

# Usamos el 70% de la base de datos como conjunto de entrenamiento y el resto como conjunto de test
sample <- sample(c(TRUE, FALSE), nrow(df3), replace=TRUE, prob=c(0.6,0.4))
datos.train  <- df3[sample, ]
datos.test   <- df3[!sample, ]

dim(datos.train)

## [1] 241  54

lr1 <- glm(final ~ Fedu , data= datos.train,family=binomial)
summary(lr1)

##
## Call:
## glm(formula = final ~ Fedu, family = binomial, data = datos.train)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.8617     0.5189   1.661  0.09679 .
## Fedu          0.6938     0.2405   2.885  0.00391 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 137.85  on 240  degrees of freedom
```

```
## Residual deviance: 128.51  on 239  degrees of freedom
## AIC: 132.51
##
## Number of Fisher Scoring iterations: 6
```

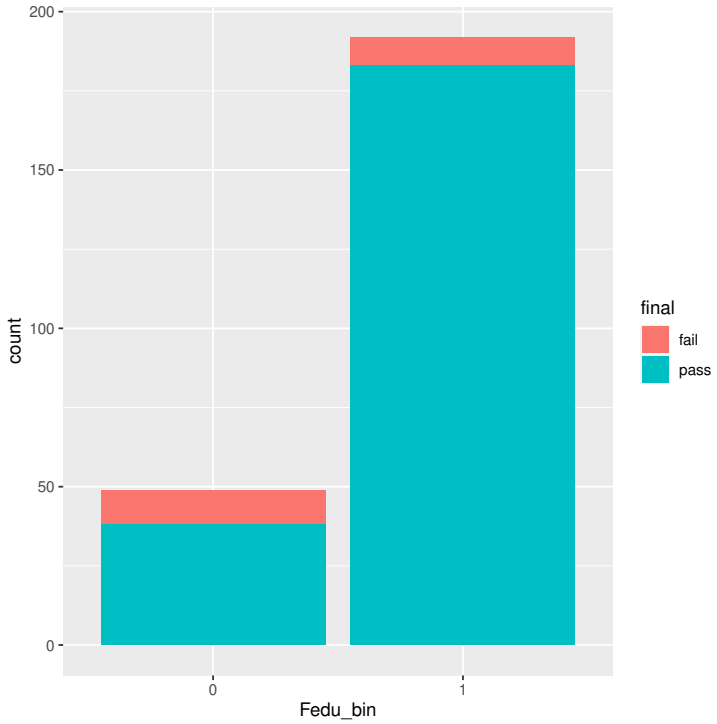
```
lr1 <- glm(final ~ as.factor(Fedu) , data= datos.train,family=binomial)
summary(lr1)
```

```
##
## Call:
## glm(formula = final ~ as.factor(Fedu), family = binomial, data = datos.train)
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)         15.57    1029.12   0.015    0.988
## as.factor(Fedu)1   -14.38    1029.12  -0.014    0.989
## as.factor(Fedu)2   -12.83    1029.12  -0.012    0.990
## as.factor(Fedu)3   -12.05    1029.12  -0.012    0.991
## as.factor(Fedu)4   -12.68    1029.12  -0.012    0.990
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 137.85  on 240  degrees of freedom
## Residual deviance: 122.94  on 236  degrees of freedom
## AIC: 132.94
##
## Number of Fisher Scoring iterations: 14
```

```
# Reagrupamos
datos.train=
  datos.train %>%
  mutate(Fedu_bin=as.factor(ifelse(Fedu>1,1,0)))

lr1 <- glm(final ~ Fedu_bin , data= datos.train,family=binomial)
summary(lr1)
```

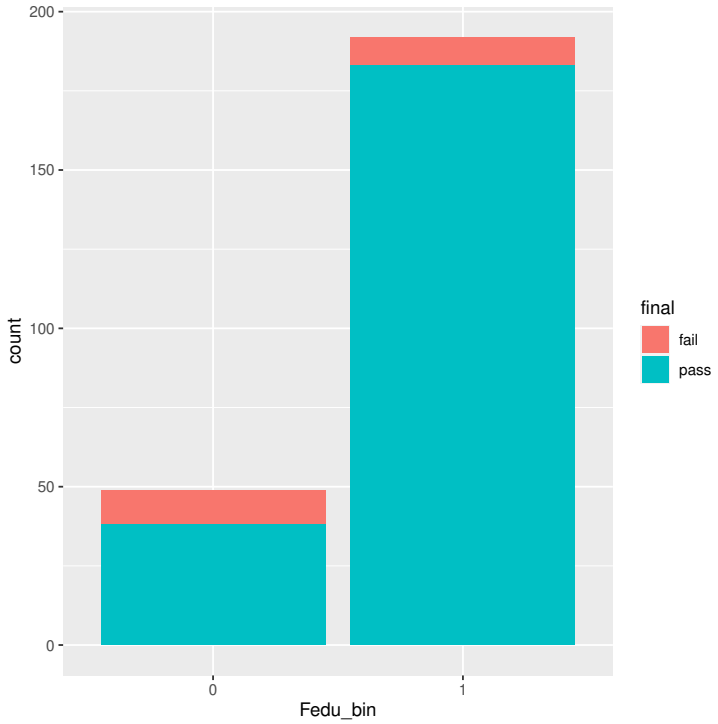
```
##
## Call:
## glm(formula = final ~ Fedu_bin, family = binomial, data = datos.train)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   1.2397     0.3424   3.621 0.000294 ***
## Fedu_bin1     1.7726     0.4835   3.666 0.000246 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 137.85  on 240  degrees of freedom
## Residual deviance: 124.84  on 239  degrees of freedom
## AIC: 128.84
##
## Number of Fisher Scoring iterations: 5
```

```
ggplot(datos.train, aes(x=Fedu_bin, group=final,fill=final)) + geom_bar()
```




```
# Medu

lr1 <- glm(final ~ Medu , data= datos.train,family=binomial)
summary(lr1)

##
## Call:
## glm(formula = final ~ Medu, family = binomial, data = datos.train)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   1.1114     0.5649   1.967   0.0492 *
## Medu          0.5079     0.2227   2.281   0.0226 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 137.85  on 240  degrees of freedom
## Residual deviance: 132.45  on 239  degrees of freedom
## AIC: 136.45
##
## Number of Fisher Scoring iterations: 5

lr1 <- glm(final ~ as.factor(Medu) , data= datos.train,family=binomial)
summary(lr1)

##
## Call:
## glm(formula = final ~ as.factor(Medu), family = binomial, data = datos.train)
```

```
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)         15.57    1455.40   0.011    0.991
## as.factor(Medu)1   -14.10    1455.40  -0.010    0.992
## as.factor(Medu)2   -13.26    1455.40  -0.009    0.993
## as.factor(Medu)3   -13.15    1455.40  -0.009    0.993
## as.factor(Medu)4   -12.31    1455.40  -0.008    0.993
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 137.85  on 240  degrees of freedom
## Residual deviance: 131.35  on 236  degrees of freedom
## AIC: 141.35
##
## Number of Fisher Scoring iterations: 14

# Aquí podríamos agrupar, o no. Agrupamos y estudiamos qué ocurre.

# Reagrupamos
datos.train=
  datos.train %>%
  mutate(Medu_bin=as.factor(ifelse(Medu>1,1,0)))

lr1 <- glm(final ~ Medu_bin , data= datos.train,family=binomial)
summary(lr1)

##
## Call:
## glm(formula = final ~ Medu_bin, family = binomial, data = datos.train)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   1.5041     0.4513   3.333 0.000861 ***
## Medu_bin1     1.1247     0.5294   2.124 0.033633 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 137.85  on 240  degrees of freedom
## Residual deviance: 133.89  on 239  degrees of freedom
## AIC: 137.89
##
## Number of Fisher Scoring iterations: 5

ggplot(datos.train, aes(x=Fedu_bin, group=final,fill=final)) + geom_bar()
```




```
# En este caso, se pierde significatividad estadística y se decide no agrupar con las mismas categorías
# sino como sigue:
datos.train=
  datos.train %>%
  mutate(Medu_bin=as.factor(ifelse(Medu==4,1,0)))

lr1 <- glm(final ~ Medu_bin , data= datos.train,family=binomial)
summary(lr1)

##
## Call:
## glm(formula = final ~ Medu_bin, family = binomial, data = datos.train)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.1296     0.2565   8.301   <2e-16 ***
## Medu_bin1     1.1285     0.6418   1.758   0.0787 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 137.85  on 240  degrees of freedom
## Residual deviance: 134.02  on 239  degrees of freedom
## AIC: 138.02
##
## Number of Fisher Scoring iterations: 6

# LR
```

```r
lr1 <- glm(final ~ Fedu_bin+Medu_bin, data= datos.train,family=binomial)
summary(lr1)
```

```
##
## Call:
## glm(formula = final ~ Fedu_bin + Medu_bin, family = binomial,
##     data = datos.train)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   1.2315     0.3427   3.594 0.000326 ***
## Fedu_bin1     1.6130     0.5299   3.044 0.002333 **
## Medu_bin1     0.4561     0.7082   0.644 0.519549
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 137.85  on 240  degrees of freedom
## Residual deviance: 124.41  on 238  degrees of freedom
## AIC: 130.41
##
## Number of Fisher Scoring iterations: 6
```

```r
# Modelo básico
base.mod <- glm(final ~ 1 , data= datos.train,family=binomial)

# Modelo completo
all.mod <- glm(final ~ Fedu_bin+Medu_bin+age+sex+school+famsize+Mjob+Fjob+reason , data= datos.train,fam

# Step-wise
stepMod <- step(base.mod, scope = list(lower = base.mod, upper = all.mod), direction = "both", trace = 0

# Variables en el modelo
formula(stepMod)
```

```
## final ~ Fedu_bin + sex + school
```

```r
# Construcción del modelo
set.seed(1337)

#  10-fold cross validation
train_control <- trainControl(method="cv", number=10)

# Entrenamos el modelo empleando glm
model <- train(formula(stepMod), data = datos.train, method = "glm",trControl=train_control,family = bin

# Resumen del modelo
summary(model)
```

```
##
## Call:
## NULL
##
## Coefficients:
```

```
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.4959     0.5815   4.292 1.77e-05 ***
## Fedu_bin1     1.6921     0.5135   3.295 0.000984 ***
## sexM         -1.4388     0.5751  -2.502 0.012360 *
## schoolMS     -1.5843     0.5832  -2.717 0.006597 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 137.85  on 240  degrees of freedom
## Residual deviance: 111.62  on 237  degrees of freedom
## AIC: 119.62
##
## Number of Fisher Scoring iterations: 6
```

```
# Evaluación del modelo
datos.test=
  datos.test %>%
  mutate(Fedu_bin=as.factor(ifelse(Fedu>1,1,0)), Medu_bin=as.factor(ifelse(Medu==4,1,0)))


prediction <- predict(model, newdata = datos.test, type = "raw")
confusionMatrix(table(prediction, datos.test$final), positive = "pass")
```

```
## Confusion Matrix and Statistics
##
##
## prediction fail pass
##       fail    0    1
##       pass   12  128
##
##                Accuracy : 0.9078
##                  95% CI : (0.8475, 0.95)
##     No Information Rate : 0.9149
##     P-Value [Acc > NIR] : 0.686406
##
##                   Kappa : -0.0133
##
##  Mcnemar's Test P-Value : 0.005546
##
##             Sensitivity : 0.9922
##             Specificity : 0.0000
##          Pos Pred Value : 0.9143
##          Neg Pred Value : 0.0000
##              Prevalence : 0.9149
##          Detection Rate : 0.9078
##    Detection Prevalence : 0.9929
##       Balanced Accuracy : 0.4961
##
##        'Positive' Class : pass
##
```

Información de la sesión de R (incluyendo información sobre el sistema operativo, la versión de R y los

paquetes usados):

```
sessionInfo()
```

```
## R version 4.3.1 (2023-06-16)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 20.04.6 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/atlas/libblas.so.3.10.3
## LAPACK: /usr/lib/x86_64-linux-gnu/atlas/liblapack.so.3.10.3;  LAPACK version 3.9.0
##
## locale:
##  [1] LC_CTYPE=es_ES.UTF-8       LC_NUMERIC=C               LC_TIME=es_ES.UTF-8
##  [4] LC_COLLATE=es_ES.UTF-8     LC_MONETARY=es_ES.UTF-8    LC_MESSAGES=es_ES.UTF-8
##  [7] LC_PAPER=es_ES.UTF-8       LC_NAME=C                  LC_ADDRESS=C
## [10] LC_TELEPHONE=C             LC_MEASUREMENT=es_ES.UTF-8 LC_IDENTIFICATION=C
##
## time zone: Europe/Madrid
## tzcode source: system (glibc)
##
## attached base packages:
## [1] grid      stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] randomForestExplainer_0.10.1 partykit_1.2-20
##  [3] mvtnorm_1.2-3                libcoin_1.0-10
##  [5] blorr_0.3.0                  Hmisc_5.1-1
##  [7] readr_2.1.4                  caretEnsemble_2.0.3
##  [9] DALEX_2.4.3                  ROCR_1.0-11
## [11] randomForest_4.7-1.1         arulesViz_1.5-2
## [13] arules_1.7-6                 Matrix_1.6-1.1
## [15] liver_1.15                   ggfortify_0.4.16
## [17] factoextra_1.0.7             mlbench_2.1-3.1
## [19] readxl_1.4.3                 caret_6.0-94
## [21] lattice_0.21-9               ggplot2_3.4.3
## [23] rpart.plot_3.1.1             rpart_4.1.19
## [25] caTools_1.18.2               dplyr_1.1.3
## [27] ISLR2_1.3-2
##
## loaded via a namespace (and not attached):
##   [1] RColorBrewer_1.1-3   rstudioapi_0.15.0    jsonlite_1.8.7       magrittr_2.0.3
##   [5] farver_2.1.1         rmarkdown_2.25       vctrs_0.6.3          base64enc_0.1-3
##   [9] iBreakDown_2.0.1     tinytex_0.47         htmltools_0.5.6.1    cellranger_1.1.0
##  [13] Formula_1.2-5        pROC_1.18.4          parallelly_1.36.0    htmlwidgets_1.6.2
##  [17] plyr_1.8.9           lubridate_1.9.3      igraph_1.5.1         lifecycle_1.0.3
##  [21] iterators_1.0.14     pkgconfig_2.0.3      R6_2.5.1             fastmap_1.1.1
##  [25] future_1.33.0        digest_0.6.33        reshape_0.8.9        GGally_2.1.2
##  [29] colorspace_2.1-0     labeling_0.4.3       fansi_1.0.5          timechange_0.2.0
##  [33] abind_1.4-5          polyclip_1.10-6      compiler_4.3.1       proxy_0.4-27
##  [37] bit64_4.0.5          withr_2.5.1          htmlTable_2.4.1      backports_1.4.1
##  [41] carData_3.0-5        viridis_0.6.4        highr_0.10           ggforce_0.4.1
##  [45] MASS_7.3-60          lava_1.7.2.1         ModelMetrics_1.2.2.2 tools_4.3.1
##  [49] foreign_0.8-85       future.apply_1.11.0  nnet_7.3-19          glue_1.6.2
##  [53] inum_1.0-5           nlme_3.1-163         checkmate_2.2.0      cluster_2.1.4
```

```
##  [57] reshape2_1.4.4       generics_0.1.3       recipes_1.0.8        gtable_0.3.4
##  [61] tzdb_0.4.0           class_7.3-22         tidyr_1.3.0          data.table_1.14.8
##  [65] hms_1.1.3            car_3.1-2            tidygraph_1.2.3      utf8_1.2.3
##  [69] ggrepel_0.9.3        foreach_1.5.2        pillar_1.9.0         stringr_1.5.0
##  [73] vroom_1.6.4          splines_4.3.1        tweenr_2.0.2         survival_3.5-7
##  [77] bit_4.0.5            tidyselect_1.2.0     pbapply_1.7-2        knitr_1.44
##  [81] gridExtra_2.3        stats4_4.3.1         xfun_0.40            graphlayouts_1.0.1
##  [85] hardhat_1.3.0        timeDate_4022.108    DT_0.30              visNetwork_2.1.2
##  [89] stringi_1.7.12       yaml_2.3.7           evaluate_0.22        codetools_0.2-19
##  [93] ggraph_2.1.0         tibble_3.2.1         cli_3.6.1            munsell_0.5.0
##  [97] Rcpp_1.0.11          globals_0.16.2       parallel_4.3.1       ellipsis_0.3.2
## [101] gower_1.0.1          bitops_1.0-7         listenv_0.9.0        viridisLite_0.4.2
## [105] ipred_0.9-14         scales_1.2.1         prodlim_2023.08.28   e1071_1.7-13
## [109] purrr_1.0.2          crayon_1.5.2         rlang_1.1.1
```

```
Sys.time()
```

```
## [1] "2023-11-03 11:55:32 CET"
```

Aprendizaje Automático I
Grado en Ciencia e Ingeniería de Datos
Universidad Rey Juan Carlos

Ejercicio: Detección Churn ACME Telephone

DSLAB

diciembre, 2023

En este ejemplo, vamos a trabajar con el dataset ACMETelephoneABT.csv (Kelleher, Namee, and D'Arcy 2015). Pasos:

1. Limpieza de los datos.
2. Dividimos los datos en 70% para grupo de training y 30% para grupo de testing.
3. Sobre el dataset de training:
    1. Transformar las variables, imputar datos (si procede).
    2. Proponer un modelo inicial (e.g. regresión logística).
    3. Evaluar inicialmente el modelo mediante los resultados básicos ofrecidos por summary(model).
4. Sobre el dataset de testing:
    1. Transformar las variables, imputar datos (si procede), siguiendo las mismas operaciones que para el caso de training.
    2. Usar el modelo propuesto para predecir si el cliente se va o no de la compañía, usando para ello el dataset de testing.
    3. Comparar con los datos de la columna real churn en los datos de testing, para comprobar el porcentaje de aciertos/fallos.
5. Decidir siguiendo criterios contrastados (con la teoría vista hasta el momento), qué variables incluimos en el modelo predictivo y cuáles no.

December 15, 2023

Solución ejercicio: Detección Churn ACME Telephone

Los siguientes resultados han sido obtenidos con un script de R.

```r
# 1. Lectura y preparación de datos

# Lectura de datos
# Strings como factores
library(readr)
library(Hmisc)
ACMETelephoneABT <- read_csv("./datos/ACMETelephoneABT.csv", na = c("", " "))

## Rows:  10000 Columns:  33
## - Column specification --------------------------------
## Delimiter:  ","
## chr  (5):  occupation, regionType, marriageStatus, creditRating, creditCard
## dbl (24):  customer, age, income, numHandsets, handsetAge, currentHandsetPrice, avgBill...
## lgl  (4):  children, smartPhone, homeOwner, churn
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.

# Corregir NAs y unificar valores en regionType
ACMETelephoneABT$regionType[which(ACMETelephoneABT$regionType == "unknown")] <- NA
ACMETelephoneABT$regionType[which(ACMETelephoneABT$regionType == "r")] <- "RURAL"
ACMETelephoneABT$regionType[which(ACMETelephoneABT$regionType == "s")] <- "SUBURBAN"
ACMETelephoneABT$regionType[which(ACMETelephoneABT$regionType == "t")] <- "TOWN"
ACMETelephoneABT$regionType = factor(ACMETelephoneABT$regionType, levels = c("RURAL", "SUBURBAN", "TOWN"

# Corregir NAs en marriageStatus
ACMETelephoneABT$marriageStatus[which(ACMETelephoneABT$marriageStatus == "unknown")] <- NA
ACMETelephoneABT$marriageStatus = factor(ACMETelephoneABT$marriageStatus, levels = c("YES", "NO"))

# Corregir NAs y unificar valores en creditCard
ACMETelephoneABT$creditCard[which(ACMETelephoneABT$creditCard == "f")] <- "FALSE"
ACMETelephoneABT$creditCard[which(ACMETelephoneABT$creditCard == "no")] <- "FALSE"
ACMETelephoneABT$creditCard[which(ACMETelephoneABT$creditCard == "t")] <- "TRUE"
ACMETelephoneABT$creditCard[which(ACMETelephoneABT$creditCard == "yes")] <- "TRUE"
ACMETelephoneABT$creditCard = factor(ACMETelephoneABT$creditCard, levels = c("TRUE", "FALSE"))

# Asignar NAs a casos con edad = 0
ACMETelephoneABT$age[which(ACMETelephoneABT$age == 0)] <- NA

# Asumimos casos de income = 0 como NAs
ACMETelephoneABT$income[which(ACMETelephoneABT$income == 0)] <- NA
```

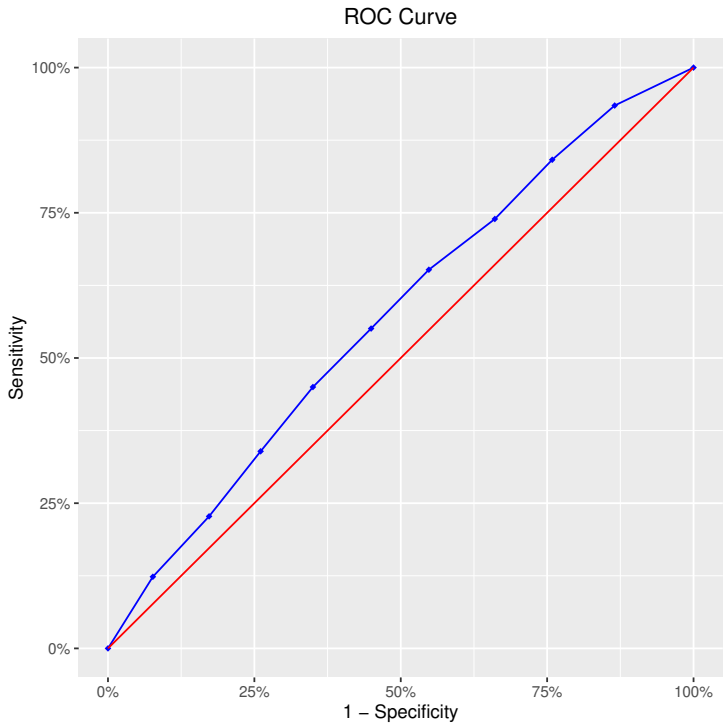
```r
levels(ACMETelephoneABT$creditCard)

## [1] "TRUE"  "FALSE"

levels(ACMETelephoneABT$regionType)

## [1] "RURAL"    "SUBURBAN" "TOWN"

levels(ACMETelephoneABT$marriageStatus)

## [1] "YES" "NO"

ACMETelephoneABT$churn = ifelse(ACMETelephoneABT$churn == "TRUE", 1, 0)
ACMETelephoneABT$churn = factor(ACMETelephoneABT$churn, levels = c(1,0))
summary(ACMETelephoneABT$churn)

##    1    0
## 5000 5000

levels(ACMETelephoneABT$churn)

## [1] "1" "0"

# 2. División de datos
library(caret)
library(dplyr)

set.seed(12345)
inTraining <- createDataPartition(pull(ACMETelephoneABT, churn),
                                  p = .7, list = FALSE, times = 1)
acme_training <- slice(ACMETelephoneABT, inTraining)
acme_testing <- slice(ACMETelephoneABT, -inTraining)

# 3. Modelo 1. Regresión Logística
min_overbundlemins = min(acme_training$avgOverBundleMins)
min_handsetAge = min(acme_training$handsetAge)
acme_training <- acme_training %>%
  mutate(binary_billAmountChangePct = ifelse(billAmountChangePct > 0, "positive","negative"))
acme_training <- acme_training %>%
  mutate(creditRating_DE = ifelse(creditRating %in% c("D", "E"), "yes","no"))
acme_training$creditRating_DE = as.factor(acme_training$creditRating_DE )

acme_training$creditRating = as.factor(acme_training$creditRating)
acme_training$binary_billAmountChangePct = as.factor(acme_training$binary_billAmountChangePct)
acme_training$homeOwner = as.factor(acme_training$homeOwner)
acme_training$smartPhone = as.factor(acme_training$smartPhone)

glm_model_train = glm(churn ~
                        log(lastMonthCustomerCareCalls + 1) +
                        log(avgrecurringCharge + 1) + log(peakOffPeakRatio + 1) +
                        log(avgBill + 1) + log(avgReceivedMins + 1) +
                        creditRating_DE + binary_billAmountChangePct + smartPhone,
                      data=acme_training, family= binomial)
summary(glm_model_train)
```

```
##
## Call:
## glm(formula = churn ~ log(lastMonthCustomerCareCalls + 1) + log(avgrecurringCharge +
##     1) + log(peakOffPeakRatio + 1) + log(avgBill + 1) + log(avgReceivedMins +
##     1) + creditRating_DE + binary_billAmountChangePct + smartPhone,
##     family = binomial, data = acme_training)
##
## Coefficients:
##                                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)                       -1.03816    0.17953  -5.783 7.36e-09 ***
## log(lastMonthCustomerCareCalls + 1)  0.10947    0.03471   3.154  0.00161 **
## log(avgrecurringCharge + 1)          0.49064    0.06612   7.421 1.16e-13 ***
## log(peakOffPeakRatio + 1)            0.06740    0.04217   1.598  0.10997
## log(avgBill + 1)                  -0.39265    0.06696  -5.864 4.52e-09 ***
## log(avgReceivedMins + 1)             0.02763    0.01592   1.735  0.08273 .
## creditRating_DEyes                   0.24202    0.06106   3.964 7.38e-05 ***
## binary_billAmountChangePctpositive   0.11028    0.05200   2.121  0.03394 *
## smartPhoneTRUE                       0.48571    0.08493   5.719 1.07e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 9704.1  on 6999  degrees of freedom
## Residual deviance: 9542.9  on 6991  degrees of freedom
## AIC: 9560.9
##
## Number of Fisher Scoring iterations: 4
```

```
# 3.1. Predicción sobre datos de test. Evaluación del modelo
min_overbundlemins = min(acme_testing$avgOverBundleMins)
min_handsetAge = min(acme_testing$handsetAge)
acme_testing <- acme_testing %>%
  mutate(binary_billAmountChangePct = ifelse(billAmountChangePct > 0, "positive","negative"))

acme_testing$income = as.factor(acme_testing$income)
acme_testing$binary_billAmountChangePct = as.factor(acme_testing$binary_billAmountChangePct)
acme_testing$homeOwner = as.factor(acme_testing$homeOwner)
acme_testing$smartPhone = as.factor(acme_testing$smartPhone)
acme_testing <- acme_testing %>%
  mutate(creditRating_DE = ifelse(creditRating %in% c("D", "E"), "yes","no"))
acme_testing$creditRating_DE = as.factor(acme_testing$creditRating_DE )

glm_probs = predict(glm_model_train, newdata = acme_testing, type = "response")

umbral_dec = 0.46
glm_probs <- ifelse(glm_probs >= umbral_dec, 1, 0)
glm_probs <- factor(glm_probs, levels = c(1,0))

tabla_conf <- table(glm_probs, acme_testing$churn)
tabla_conf
```

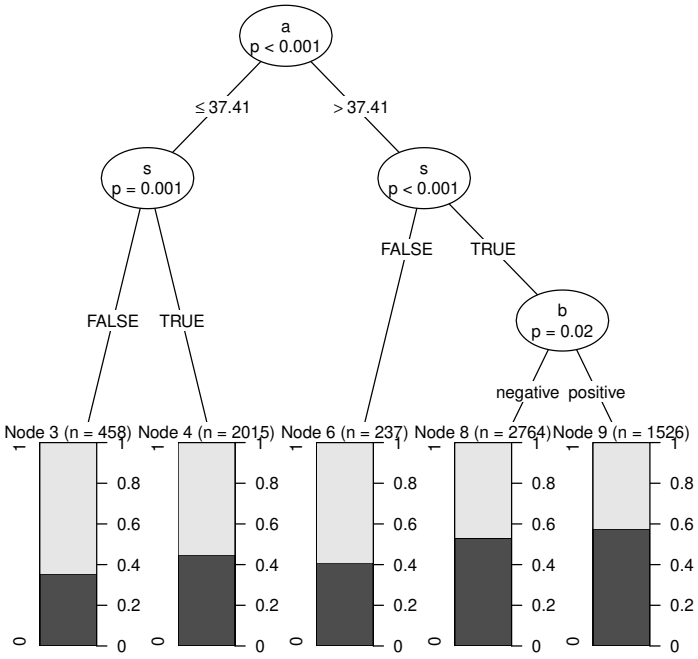
```
##
## glm_probs    1    0
```

```
##          1 1075 1200
##          0  425  300

caret::confusionMatrix(tabla_conf, positive = '1')

## Confusion Matrix and Statistics
##
##
## glm_probs    1    0
##          1 1075 1200
##          0  425  300
##
##                Accuracy : 0.4583
##                  95% CI : (0.4404, 0.4764)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : -0.0833
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.7167
##             Specificity : 0.2000
##          Pos Pred Value : 0.4725
##          Neg Pred Value : 0.4138
##              Prevalence : 0.5000
##          Detection Rate : 0.3583
##    Detection Prevalence : 0.7583
##       Balanced Accuracy : 0.4583
##
##        'Positive' Class : 1
##

# Curva ROC
logistic_gains_table <- blr_gains_table(glm_model_train, data = acme_testing)
blr_roc_curve(logistic_gains_table)
```

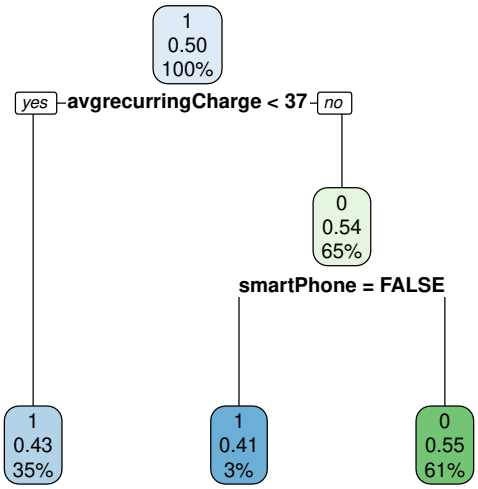



```
# 4. Modelo 2. Árbol de decisión
library(partykit)
ctree_acme = ctree(churn ~ avgOverBundleMins +
                     lastMonthCustomerCareCalls +
                     avgrecurringCharge + peakOffPeakRatio +
                     binary_billAmountChangePct + smartPhone,
                   data=acme_training)
ctree_acme

##
## Model formula:
## churn ~ avgOverBundleMins + lastMonthCustomerCareCalls + avgrecurringCharge +
##     peakOffPeakRatio + binary_billAmountChangePct + smartPhone
##
## Fitted party:
## [1] root
## |   [2] avgrecurringCharge <= 37.41
## |   |   [3] smartPhone in FALSE: 1 (n = 458, err = 35.4%)
## |   |   [4] smartPhone in TRUE: 1 (n = 2015, err = 44.8%)
## |   [5] avgrecurringCharge > 37.41
## |   |   [6] smartPhone in FALSE: 1 (n = 237, err = 40.9%)
## |   |   [7] smartPhone in TRUE
## |   |   |   [8] binary_billAmountChangePct in negative: 0 (n = 2764, err = 47.1%)
## |   |   |   [9] binary_billAmountChangePct in positive: 0 (n = 1526, err = 42.5%)
##
## Number of inner nodes:    4
## Number of terminal nodes: 5

plot(ctree_acme, gp = gpar(fontsize = 10),
     inner_panel=node_inner,
```

```
    ip_args=list(
      abbreviate = TRUE,
      id = FALSE)
)
```




```
library(rpart)
library(rpart.plot)
rpart_acme = rpart(churn ~ avgOverBundleMins +
                     lastMonthCustomerCareCalls +
                     avgrecurringCharge + peakOffPeakRatio +
                     binary_billAmountChangePct + smartPhone,
                   data=acme_training)
rpart_acme

## n= 7000
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 7000 3500 1 (0.5000000 0.5000000)
##   2) avgrecurringCharge< 37.43 2473 1064 1 (0.5697533 0.4302467) *
##   3) avgrecurringCharge>=37.43 4527 2091 0 (0.4618953 0.5381047)
##     6) smartPhone=FALSE 237   97 1 (0.5907173 0.4092827) *
##     7) smartPhone=TRUE 4290 1951 0 (0.4547786 0.5452214) *

rpart.plot(rpart_acme)
```




```
# 4.2 Predicción sobre datos de test
ctree_pred <- predict(ctree_acme, newdata=acme_testing, type='response')
confusionMatrix(ctree_pred, acme_testing$churn)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   0
##          1 629 522
##          0 871 978
##
##                Accuracy : 0.5357
##                  95% CI : (0.5176, 0.5536)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : 5.005e-05
##
##                   Kappa : 0.0713
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4193
##             Specificity : 0.6520
##          Pos Pred Value : 0.5465
##          Neg Pred Value : 0.5289
##              Prevalence : 0.5000
##          Detection Rate : 0.2097
##    Detection Prevalence : 0.3837
##       Balanced Accuracy : 0.5357
##
##        'Positive' Class : 1
```

```
##
```

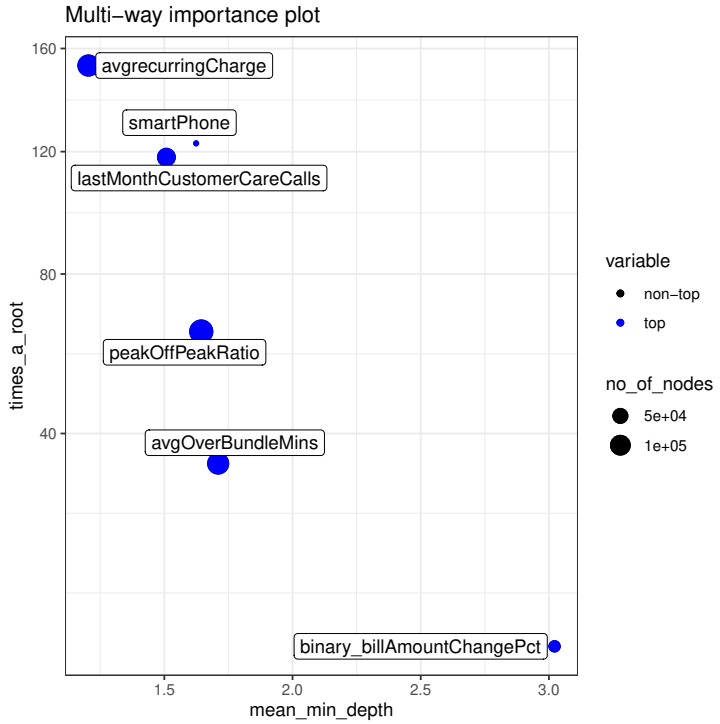
```
rpart_pred <- predict(rpart_acme, newdata=acme_testing, type='class')
confusionMatrix(rpart_pred, acme_testing$churn)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   0
##          1 629 522
##          0 871 978
##
##                Accuracy : 0.5357
##                  95% CI : (0.5176, 0.5536)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : 5.005e-05
##
##                   Kappa : 0.0713
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4193
##             Specificity : 0.6520
##          Pos Pred Value : 0.5465
##          Neg Pred Value : 0.5289
##              Prevalence : 0.5000
##          Detection Rate : 0.2097
##    Detection Prevalence : 0.3837
##       Balanced Accuracy : 0.5357
##
##        'Positive' Class : 1
##
```

```
# 5. Modelo 3: Random Forest
library(randomForest)

forest_acme = randomForest(churn ~ avgOverBundleMins +
                             lastMonthCustomerCareCalls +
                             avgrecurringCharge + peakOffPeakRatio +
                             binary_billAmountChangePct + smartPhone,
                           data=acme_training)
forest_acme
```

```
##
## Call:
##  randomForest(formula = churn ~ avgOverBundleMins + lastMonthCustomerCareCalls +      avgrecurringCha
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 45.7%
## Confusion matrix:
##      1    0 class.error
## 1 1797 1703   0.4865714
## 0 1496 2004   0.4274286
```

```
library(randomForestExplainer)
importance_frame <- measure_importance(forest_acme)

## [1] "Warning: your forest does not contain information on local importance so 'accuracy_decrease' mea

save(importance_frame, file = "importance_frame.rda")
load("importance_frame.rda")
plot_multi_way_importance(importance_frame, size_measure = "no_of_nodes")
```



Información de la sesión de R (incluyendo información sobre el sistema operativo, la versión de R y los paquetes usados):

```
sessionInfo()

## R version 4.3.1 (2023-06-16)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 20.04.6 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/atlas/libblas.so.3.10.3
## LAPACK: /usr/lib/x86_64-linux-gnu/atlas/liblapack.so.3.10.3;  LAPACK version 3.9.0
##
## locale:
##  [1] LC_CTYPE=es_ES.UTF-8       LC_NUMERIC=C               LC_TIME=es_ES.UTF-8
##  [4] LC_COLLATE=es_ES.UTF-8     LC_MONETARY=es_ES.UTF-8    LC_MESSAGES=es_ES.UTF-8
##  [7] LC_PAPER=es_ES.UTF-8       LC_NAME=C                  LC_ADDRESS=C
## [10] LC_TELEPHONE=C             LC_MEASUREMENT=es_ES.UTF-8 LC_IDENTIFICATION=C
##
## time zone: Europe/Madrid
## tzcode source: system (glibc)
##
```

```
## attached base packages:
## [1] grid      stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] randomForestExplainer_0.10.1 partykit_1.2-20
##  [3] mvtnorm_1.2-3                libcoin_1.0-10
##  [5] blorr_0.3.0                  Hmisc_5.1-1
##  [7] readr_2.1.4                  caretEnsemble_2.0.3
##  [9] DALEX_2.4.3                  ROCR_1.0-11
## [11] randomForest_4.7-1.1         arulesViz_1.5-2
## [13] arules_1.7-6                 Matrix_1.6-1.1
## [15] liver_1.15                   ggfortify_0.4.16
## [17] factoextra_1.0.7             mlbench_2.1-3.1
## [19] readxl_1.4.3                 caret_6.0-94
## [21] lattice_0.21-9               ggplot2_3.4.3
## [23] rpart.plot_3.1.1             rpart_4.1.19
## [25] caTools_1.18.2               dplyr_1.1.3
## [27] ISLR2_1.3-2
##
## loaded via a namespace (and not attached):
##   [1] RColorBrewer_1.1-3   rstudioapi_0.15.0    jsonlite_1.8.7       magrittr_2.0.3
##   [5] farver_2.1.1         rmarkdown_2.25       vctrs_0.6.3          base64enc_0.1-3
##   [9] iBreakDown_2.0.1     tinytex_0.47         htmltools_0.5.6.1    cellranger_1.1.0
##  [13] Formula_1.2-5        pROC_1.18.4          parallelly_1.36.0    htmlwidgets_1.6.2
##  [17] plyr_1.8.9           lubridate_1.9.3      igraph_1.5.1         lifecycle_1.0.3
##  [21] iterators_1.0.14     pkgconfig_2.0.3      R6_2.5.1             fastmap_1.1.1
##  [25] future_1.33.0        digest_0.6.33        reshape_0.8.9        GGally_2.1.2
##  [29] colorspace_2.1-0     labeling_0.4.3       fansi_1.0.5          timechange_0.2.0
##  [33] abind_1.4-5          polyclip_1.10-6      compiler_4.3.1       proxy_0.4-27
##  [37] bit64_4.0.5          withr_2.5.1          htmlTable_2.4.1      backports_1.4.1
##  [41] carData_3.0-5        viridis_0.6.4        highr_0.10           ggforce_0.4.1
##  [45] MASS_7.3-60          lava_1.7.2.1         ModelMetrics_1.2.2.2 tools_4.3.1
##  [49] foreign_0.8-85       future.apply_1.11.0  nnet_7.3-19          glue_1.6.2
##  [53] inum_1.0-5           nlme_3.1-163         checkmate_2.2.0      cluster_2.1.4
##  [57] reshape2_1.4.4       generics_0.1.3       recipes_1.0.8        gtable_0.3.4
##  [61] tzdb_0.4.0           class_7.3-22         tidyr_1.3.0          data.table_1.14.8
##  [65] hms_1.1.3            car_3.1-2            tidygraph_1.2.3      utf8_1.2.3
##  [69] ggrepel_0.9.3        foreach_1.5.2        pillar_1.9.0         stringr_1.5.0
##  [73] vroom_1.6.4          splines_4.3.1        tweenr_2.0.2         survival_3.5-7
##  [77] bit_4.0.5            tidyselect_1.2.0     pbapply_1.7-2        knitr_1.44
##  [81] gridExtra_2.3        stats4_4.3.1         xfun_0.40            graphlayouts_1.0.1
##  [85] hardhat_1.3.0        timeDate_4022.108    DT_0.30              visNetwork_2.1.2
##  [89] stringi_1.7.12       yaml_2.3.7           evaluate_0.22        codetools_0.2-19
##  [93] ggraph_2.1.0         tibble_3.2.1         cli_3.6.1            munsell_0.5.0
##  [97] Rcpp_1.0.11          globals_0.16.2       parallel_4.3.1       ellipsis_0.3.2
## [101] gower_1.0.1          bitops_1.0-7         listenv_0.9.0        viridisLite_0.4.2
## [105] ipred_0.9-14         scales_1.2.1         prodlim_2023.08.28   e1071_1.7-13
## [109] purrr_1.0.2          crayon_1.5.2         rlang_1.1.1

Sys.time()

## [1] "2023-11-02 21:54:13 CET"
```

Aprendizaje Automático I
Grado en Ciencia e Ingeniería de Datos
Universidad Rey Juan Carlos

Ejercicio: Reglas de Asociación

DSLab

diciembre, 2023

Considera la siguiente lista de compras:

```
# Lista de compras
market_basket <-
  list(
  c("apple", "beer", "rice", "meat"),
  c("apple", "beer", "rice"),
  c("apple", "beer", "rice"),
  c("apple", "beer"),
  c("apple", "pear"),
  c("apple", "beer", "rice", "pear"),
  c("milk", "beer", "rice", "meat"),
  c("apple", "beer", "rice"),
  c("apple", "rice", "pear"),
  c("milk", "beer", "rice", "meat"),
  c("milk", "rice"),
  c("apple", "beer"),
  c("milk", "rice"),
  c("milk", "beer"),
  c("milk", "pear")
  )
```

```
# nombramos las compras (C1 a C10)
names(market_basket) <- paste("C", c(1:8), sep = "")
```

1. Carga los datos y explora su contenido. Importa las librerías `arules` y `arulesViz`. Carga los datos de la compra en un objeto de la "clase transacción" para poder analizar los datos. Para ello se utiliza la siguiente función del paquete arules:

```
 library(arules)

trans <- as(market_basket, "transactions")
```

2. Obten una lista de los productos en la cesta de la compra empleando la función `itemLabels`.
3. Como hay pocas transacciones puedes visualizarlas con la función `image`. ¿Qué porcentaje de celdas de la imagen está vacías?
4. Mediante la función `itemFrequencyPlot` puedes visualizar la frecuencia relativa de los productos.
5. Emplea la función `apriori` para analizar las reglas. Emplea `supp=0.3` y `conf=0.5`.
6. Inspecciona las reglas mediante la función `inspect`. ¿qué regla, con al menos dos productos, se encuentra en el mayor número de compras? ¿Qué confianza se tiene en que quien compra `apple` también incluya `rice` en su compra?
7. Emplea la siguiente función para averiguar qué compran los clientes además de comprar `beer`:

```
beer_rules_lhs <- apriori(trans,
                          parameter = list(supp=0.3, conf=0.5,
                                           maxlen=10,
                                           minlen=2),
                          appearance = list(lhs="beer", default="rhs"))
```

8. Visualiza las reglas. Las técnicas basadas en grafos se centran en la relación entre los distintos elementos del conjunto de reglas. Representan las reglas (o conjuntos de reglas) como un gráfico en el que los elementos son vértices etiquetados y las reglas (o conjuntos de reglas) se representan como vértices conectados a los elementos mediante flechas. Emplea las siguientes instrucciones para obtener un gráfico de red que muestra las asociaciones entre los elementos seleccionados. Los círculos más grandes implican mayor apoyo, mientras que los círculos rojos implican mayor elevación.

```
subrules \<- head(rules, n=10, by = "confidence")
plot(subrules, method = "graph", engine = "htmlwidget")
```

9. Intepreta el siguiente gráfico:

```
plot(subrules, method="paracoord")
```

December 15, 2023

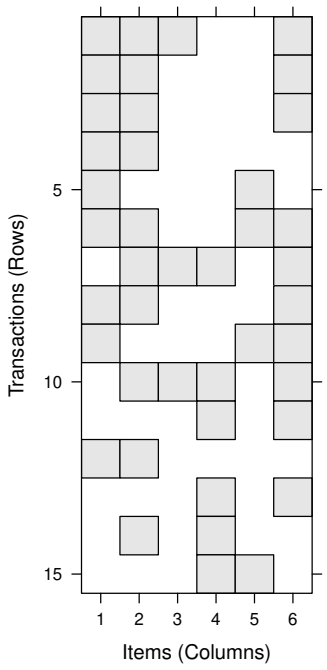
Solución ejercicio: Reglas de asociación

Los siguientes resultados han sido obtenidos con un script de R.

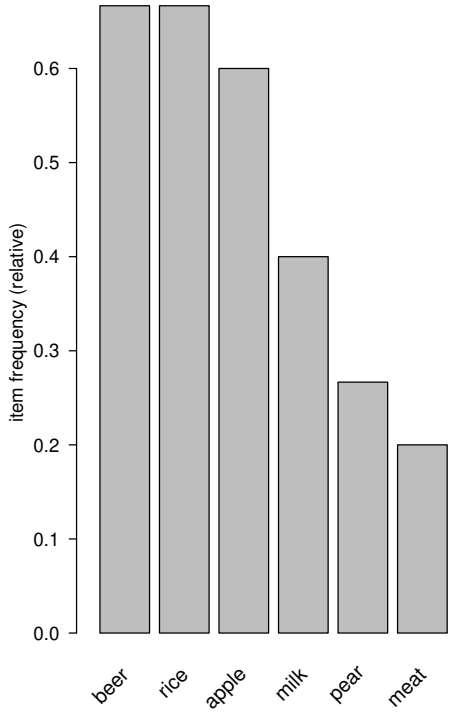
```r
# Liberías necesarias para resolver el ejercicio
library(arules)
library(arulesViz)
# Lista de compras
market_basket <-
  list(
    c("apple", "beer", "rice", "meat"),
    c("apple", "beer", "rice"),
    c("apple", "beer", "rice"),
    c("apple", "beer"),
    c("apple", "pear"),
    c("apple", "beer", "rice", "pear"),
    c("milk", "beer", "rice", "meat"),
    c("apple", "beer", "rice"),
    c("apple", "rice", "pear"),
    c("milk", "beer", "rice", "meat"),
    c("milk", "rice"),
    c("apple", "beer"),
    c("milk", "rice"),
    c("milk", "beer"),
    c("milk", "pear")
  )

# nombramos las compras
names(market_basket) <- paste("C", c(1:length(market_basket)), sep = "")

# Transformación
trans <- as(market_basket, "transactions")

# Lista de productos
itemLabels(trans)

## [1] "apple" "beer"  "meat"  "milk"  "pear"  "rice"

# Resumen de los datos
summary(trans)

## transactions as itemMatrix in sparse format with
##  15 rows (elements/itemsets/transactions) and
##  6 columns (items) and a density of 0.4666667
##
## most frequent items:
```

```
##    beer    rice   apple    milk    pear (Other)
##      10      10       9       6       4       3
##
## element (itemset/transaction) length distribution:
## sizes
## 2 3 4
## 7 4 4
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     2.0     2.0     3.0     2.8     3.5     4.0
##
## includes extended item information - examples:
##   labels
## 1  apple
## 2   beer
## 3   meat
##
## includes extended transaction information - examples:
##   transactionID
## 1            C1
## 2            C2
## 3            C3
```

```
# Visualización
image(trans)
```



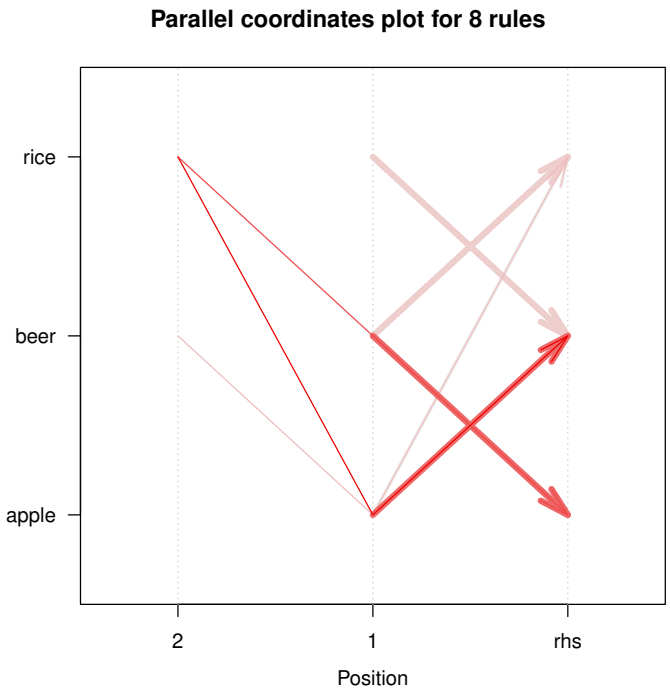

```
# Frecuencia relativa de los productos
itemFrequencyPlot(trans, topN=10,  cex.names=1)
```




```
# A-Priori

#Min Support 0.3, confidence 0.5.
rules <- apriori(trans,
                 parameter = list(supp=0.3, conf=0.5,
                                  maxlen=10,
                                  target= "rules"))

## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target
##         0.5    0.1    1 none FALSE            TRUE       5     0.3      1     10  rules
##   ext
##  TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 4
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[6 item(s), 15 transaction(s)] done [0.00s].
## sorting and recoding items ... [4 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 done [0.00s].
## writing ... [12 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].

summary(rules)
```

```
## set of 12 rules
##
## rule length distribution (lhs + rhs):sizes
## 1 2 3
## 3 6 3
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00    1.75    2.00    2.00    2.25    3.00
##
## summary of quality measures:
##     support         confidence        coverage           lift           count
##  Min.   :0.3333   Min.   :0.6000   Min.   :0.4000   Min.   :1.000   Min.   : 5.00
##  1st Qu.:0.3833   1st Qu.:0.6667   1st Qu.:0.5667   1st Qu.:1.000   1st Qu.: 5.75
##  Median :0.4667   Median :0.7000   Median :0.6667   Median :1.050   Median : 7.00
##  Mean   :0.4667   Mean   :0.6950   Mean   :0.6833   Mean   :1.079   Mean   : 7.00
##  3rd Qu.:0.5000   3rd Qu.:0.7143   3rd Qu.:0.7500   3rd Qu.:1.167   3rd Qu.: 7.50
##  Max.   :0.6667   Max.   :0.8333   Max.   :1.0000   Max.   :1.250   Max.   :10.00
##
## mining info:
##   data ntransactions support confidence
##  trans            15     0.3        0.5
##                                                                                call
##  apriori(data = trans, parameter = list(supp = 0.3, conf = 0.5, maxlen = 10, target = "rules"))

# Producto `beer`
beer_rules_lhs <- apriori(trans,
                          parameter = list(supp=0.3, conf=0.5,
                                           maxlen=10,
                                           minlen=2),
                          appearance = list(lhs="beer", default="rhs"))

## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target
##         0.5    0.1    1 none FALSE            TRUE       5     0.3      2     10  rules
##   ext
##  TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 4
##
## set item appearances ...[1 item(s)] done [0.00s].
## set transactions ...[6 item(s), 15 transaction(s)] done [0.00s].
## sorting and recoding items ... [4 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 done [0.00s].
## writing ... [2 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].

inspect(beer_rules_lhs)
```

```
##     lhs       rhs     support   confidence coverage  lift     count
## [1] {beer} => {apple} 0.4666667 0.7        0.6666667 1.166667 7
## [2] {beer} => {rice}  0.4666667 0.7        0.6666667 1.050000 7

# Visualizar las reglas
subrules <- head(rules, n=10, by = "confidence")

plot(subrules, method = "graph",  engine = "htmlwidget")

## Error:  package 'webshot' was installed before R 4.0.0:  please re-install it

plot(subrules, method="paracoord")
```



Información de la sesión de R (incluyendo información sobre el sistema operativo, la versión de R y los paquetes usados):

```
sessionInfo()

## R version 4.3.1 (2023-06-16)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 20.04.6 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/atlas/libblas.so.3.10.3
## LAPACK: /usr/lib/x86_64-linux-gnu/atlas/liblapack.so.3.10.3;  LAPACK version 3.9.0
##
## locale:
##  [1] LC_CTYPE=es_ES.UTF-8       LC_NUMERIC=C               LC_TIME=es_ES.UTF-8
##  [4] LC_COLLATE=es_ES.UTF-8     LC_MONETARY=es_ES.UTF-8    LC_MESSAGES=es_ES.UTF-8
##  [7] LC_PAPER=es_ES.UTF-8       LC_NAME=C                  LC_ADDRESS=C
## [10] LC_TELEPHONE=C             LC_MEASUREMENT=es_ES.UTF-8 LC_IDENTIFICATION=C
```

```
##
## time zone: Europe/Madrid
## tzcode source: system (glibc)
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] arulesViz_1.5-2  arules_1.7-6     Matrix_1.6-1.1   liver_1.15       ggfortify_0.4.16
##  [6] factoextra_1.0.7 mlbench_2.1-3.1  readxl_1.4.3     caret_6.0-94     lattice_0.21-9
## [11] ggplot2_3.4.3    rpart.plot_3.1.1 rpart_4.1.19     caTools_1.18.2   dplyr_1.1.3
## [16] ISLR2_1.3-2
##
## loaded via a namespace (and not attached):
##  [1] bitops_1.0-7         pROC_1.18.4          gridExtra_2.3        rlang_1.1.1
##  [5] magrittr_2.0.3       e1071_1.7-13         compiler_4.3.1       vctrs_0.6.3
##  [9] reshape2_1.4.4       stringr_1.5.0        pkgconfig_2.0.3      fastmap_1.1.1
## [13] ellipsis_0.3.2       labeling_0.4.3       ggraph_2.1.0         utf8_1.2.3
## [17] rmarkdown_2.25       prodlim_2023.08.28   tzdb_0.4.0           tinytex_0.47
## [21] purrr_1.0.2          xfun_0.40            jsonlite_1.8.7       recipes_1.0.8
## [25] highr_0.10           tweenr_2.0.2         parallel_4.3.1       R6_2.5.1
## [29] stringi_1.7.12       parallelly_1.36.0    lubridate_1.9.3      cellranger_1.1.0
## [33] Rcpp_1.0.11          iterators_1.0.14     knitr_1.44           future.apply_1.11.0
## [37] readr_2.1.4          splines_4.3.1        nnet_7.3-19          igraph_1.5.1
## [41] timechange_0.2.0     tidyselect_1.2.0     rstudioapi_0.15.0    yaml_2.3.7
## [45] viridis_0.6.4        timeDate_4022.108    codetools_0.2-19     listenv_0.9.0
## [49] tibble_3.2.1         plyr_1.8.9           withr_2.5.1          evaluate_0.22
## [53] future_1.33.0        survival_3.5-7       proxy_0.4-27         polyclip_1.10-6
## [57] pillar_1.9.0         foreach_1.5.2        stats4_4.3.1         generics_0.1.3
## [61] hms_1.1.3            munsell_0.5.0        scales_1.2.1         globals_0.16.2
## [65] class_7.3-22         glue_1.6.2           tools_4.3.1          data.table_1.14.8
## [69] ModelMetrics_1.2.2.2 gower_1.0.1          visNetwork_2.1.2     graphlayouts_1.0.1
## [73] tidygraph_1.2.3      grid_4.3.1           tidyr_1.3.0          ipred_0.9-14
## [77] colorspace_2.1-0     nlme_3.1-163         ggforce_0.4.1        cli_3.6.1
## [81] fansi_1.0.5          viridisLite_0.4.2    lava_1.7.2.1         gtable_0.3.4
## [85] digest_0.6.33        ggrepel_0.9.3        htmlwidgets_1.6.2    farver_2.1.1
## [89] htmltools_0.5.6.1    lifecycle_1.0.3      hardhat_1.3.0        MASS_7.3-60
```

```
Sys.time()
```

```
## [1] "2023-11-02 18:33:30 CET"
```

Aprendizaje Automático I
Grado en Ciencia e Ingeniería de Datos
Universidad Rey Juan Carlos

Ejercicio: Valores SHAP

DSLab

diciembre, 2023

Los modelos predictivos complejos (random forest, xgboost, deep learning) no son fáciles de interpretar.

En modelos predictivos, dada una determinada predicción, responder a esta pregunta: ¿cuál ha sido la influencia de cada variable de entrada para obtener esa predicción?, no siempre es sencillo.

Una técnica reciente para interpretar modelos complejos es: SHAP (SHapley Additive exPlanations) desarrollada por Scott M. Lundberg.

SHAP mide el impacto de las variables teniendo en cuenta la interacción con otras variables. Evalua cómo un ligero cambio en una variable puede cambiar mucho el resultado final. Los valores de Shapley calculan la importancia de una característica comparando lo que un modelo predice con y sin la característica. Sin embargo, dado que el orden en que un modelo recibe las características puede afectar a sus predicciones (piensa en un árbol de decisión), esto se hace en todos los órdenes posibles, para que las características se comparen equitativamente.

Consideremos un modelo de `random forest` entrenado sobre los datos `Caravan` de la librería `ISLR2`. Los datos contienen 5822 registros de clientes reales. Cada registro consta de 86 variables, que contienen datos sociodemográficos (variables 1-43) y propiedad de productos (variables 44-86). Los datos sociodemográficos proceden de los códigos postales. Todos los clientes que viven en zonas con el mismo código postal tienen los mismos

atributos sociodemográficos. La variable 86 (Compra) indica si el cliente ha comprado una póliza de seguro de caravana. Puede obtenerse más información sobre las distintas variables en http://www.liacs.nl/~putten/library/cc2000/data.html

1. Estudia el conjunto de datos. ¿Qué puedes concluir de los datos dada la distribución de la variable respuesta?
2. Equilibra las clases para que el 50% de las observaciones de entrenamiento tengan 'Yes' en la variable respuesta, y el restante 50% tengan 'No'.
3. Ajusta un modelo de `random forest` empleando la función `randomForest` de la librería `randomForest`. ¿Qué variables son más importantes en la construcción del bosque?
4. Construye una curva ROC para estudiar el rendimiento del modelo.
5. Empleando la función `explain` de librería `DALEX` y la visualización mediante un `plot` explica la predicción obtenida para las dos primeras observaciones de la muestra de prueba.

December 15, 2023

Solución ejercicio: Valores SHAP

Los siguientes resultados han sido obtenidos con un script de R.

```
# Liberías necesarias para resolver el ejercicio
library(ISLR2)
library(ggplot2)
library(randomForest)
library(ROCR)
library("DALEX")

# Variable respuesta a factor
car=Caravan
car$Purchase=as.factor(car$Purchase)
# Dividimos la muestra en train, test
# Parciticionamos los datos
set.seed(2138)
n=dim(car)[1]
indices=seq(1:n)
indices.train=sample(indices,size=n*.8,replace=FALSE)
indices.test=sample(indices[-indices.train],size=n*.1,replace=FALSE)
indices.valid=indices[-c(indices.train,indices.test)]

car.train=car[indices.train,]
car.test=car[indices.test,]
car.valid=car[indices.valid,]

# EDA
str(Caravan)

## 'data.frame': 5822 obs. of  86 variables:
##  $ MOSTYPE : num  33 37 37 9 40 23 39 33 33 11 ...
##  $ MAANTHUI: num  1 1 1 1 1 1 2 1 1 2 ...
##  $ MGEMOMV : num  3 2 2 3 4 2 3 2 2 3 ...
##  $ MGEMLEEF: num  2 2 2 3 2 1 2 3 4 3 ...
##  $ MOSHOOFD: num  8 8 8 3 10 5 9 8 8 3 ...
##  $ MGODRK  : num  0 1 0 2 1 0 2 0 0 3 ...
##  $ MGODPR  : num  5 4 4 3 4 5 2 7 1 5 ...
##  $ MGODOV  : num  1 1 2 2 1 0 0 0 3 0 ...
##  $ MGODGE  : num  3 4 4 4 4 5 5 2 6 2 ...
##  $ MRELGE  : num  7 6 3 5 7 0 7 7 6 7 ...
##  $ MRELSA  : num  0 2 2 2 1 6 2 2 0 0 ...
##  $ MRELOV  : num  2 2 4 2 2 3 0 0 3 2 ...
##  $ MFALLEEN: num  1 0 4 2 2 3 0 0 3 2 ...
##  $ MFGEKIND: num  2 4 4 3 4 5 3 5 3 2 ...
```

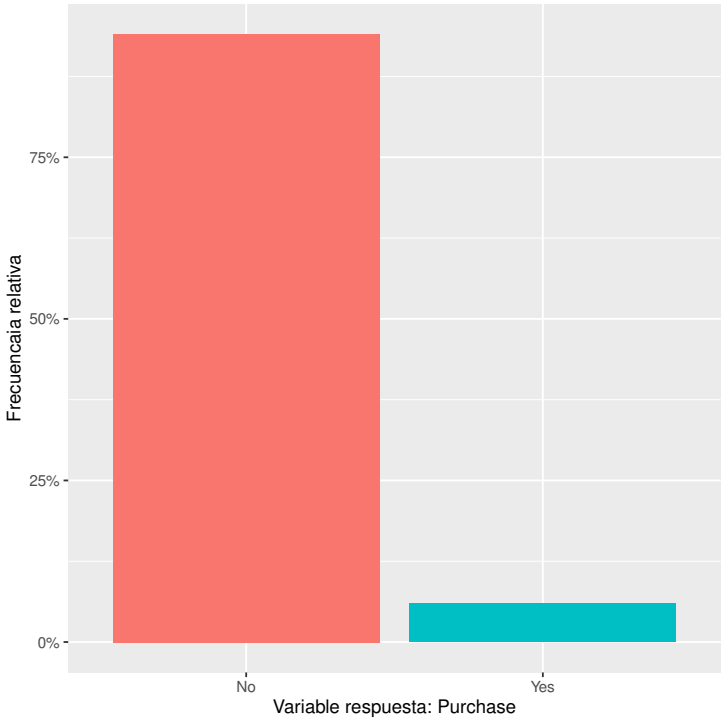
```
##  $ MFWEKIND: num  6 5 2 4 4 2 6 4 3 6 ...
##  $ MOPLHOOG: num  1 0 0 3 5 0 0 0 0 0 ...
##  $ MOPLMIDD: num  2 5 5 4 4 5 4 3 1 4 ...
##  $ MOPLLAAG: num  7 4 4 2 0 4 5 6 8 5 ...
##  $ MBERHOOG: num  1 0 0 4 0 2 0 2 1 2 ...
##  $ MBERZELF: num  0 0 0 0 5 0 0 0 1 0 ...
##  $ MBERBOER: num  1 0 0 0 4 0 0 0 0 0 ...
##  $ MBERMIDD: num  2 5 7 3 0 4 4 2 1 3 ...
##  $ MBERARBG: num  5 0 0 1 0 2 1 5 8 3 ...
##  $ MBERARBO: num  2 4 2 2 0 2 5 2 1 3 ...
##  $ MSKA    : num  1 0 0 3 9 2 0 2 1 1 ...
##  $ MSKB1   : num  1 2 5 2 0 2 1 1 1 2 ...
##  $ MSKB2   : num  2 3 0 1 0 2 4 2 0 1 ...
##  $ MSKC    : num  6 5 4 4 0 4 5 5 8 4 ...
##  $ MSKD    : num  1 0 0 0 0 2 0 2 1 2 ...
##  $ MHHUUR  : num  1 2 7 5 4 9 6 0 9 0 ...
##  $ MHKOOP  : num  8 7 2 4 5 0 3 9 0 9 ...
##  $ MAUT1   : num  8 7 7 9 6 5 8 4 5 6 ...
##  $ MAUT2   : num  0 1 0 0 2 3 0 4 2 1 ...
##  $ MAUT0   : num  1 2 2 0 1 3 1 2 3 2 ...
##  $ MZFONDS : num  8 6 9 7 5 9 9 6 7 6 ...
##  $ MZPART  : num  1 3 0 2 4 0 0 3 2 3 ...
##  $ MINKM30 : num  0 2 4 1 0 5 4 2 7 2 ...
##  $ MINK3045: num  4 0 5 5 0 2 3 5 2 3 ...
##  $ MINK4575: num  5 5 0 3 9 3 3 3 1 3 ...
##  $ MINK7512: num  0 2 0 0 0 0 0 0 0 1 ...
##  $ MINK123M: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ MINKGEM : num  4 5 3 4 6 3 3 3 2 4 ...
##  $ MKOOPKLA: num  3 4 4 4 3 3 5 3 3 7 ...
##  $ PWAPART : num  0 2 2 0 0 0 0 0 0 2 ...
##  $ PWABEDR : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ PWALAND : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ PPERSAUT: num  6 0 6 6 0 6 6 0 5 0 ...
##  $ PBESAUT : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ PMOTSCO : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ PVRAAUT : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ PAANHANG: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ PTRACTOR: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ PWERKT  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ PBROM   : num  0 0 0 0 0 0 0 3 0 0 ...
##  $ PLEVEN  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ PPERSONG: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ PGEZONG : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ PWAOREG : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ PBRAND  : num  5 2 2 2 6 0 0 0 0 3 ...
##  $ PZEILPL : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ PPLEZIER: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ PFIETS  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ PINBOED : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ PBYSTAND: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ AWAPART : num  0 2 1 0 0 0 0 0 0 1 ...
##  $ AWABEDR : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ AWALAND : num  0 0 0 0 0 0 0 0 0 0 ...
```

```
##  $ APERSAUT: num  1 0 1 1 0 1 1 0 1 0 ...
##  $ ABESAUT : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ AMOTSCO : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ AVRAAUT : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ AAANHANG: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ ATRACTOR: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ AWERKT  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ ABROM   : num  0 0 0 0 0 0 0 1 0 0 ...
##  $ ALEVEN  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ APERSONG: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ AGEZONG : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ AWAOREG : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ ABRAND  : num  1 1 1 1 1 0 0 0 0 1 ...
##  $ AZEILPL : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ APLEZIER: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ AFIETS  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ AINBOED : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ ABYSTAND: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Purchase: Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 ...

summary(car.train)

##     MOSTYPE         MAANTHUI        MGEMOMV         MGEMLEEF        MOSHOOFD
##  Min.   : 1.00   Min.   : 1.000   Min.   :1.000   Min.   :1.000   Min.   : 1.000
##  1st Qu.:10.00   1st Qu.: 1.000   1st Qu.:2.000   1st Qu.:2.000   1st Qu.: 3.000
##  Median :30.00   Median : 1.000   Median :3.000   Median :3.000   Median : 7.000
##  Mean   :24.18   Mean   : 1.111   Mean   :2.681   Mean   :2.985   Mean   : 5.756
##  3rd Qu.:35.00   3rd Qu.: 1.000   3rd Qu.:3.000   3rd Qu.:3.000   3rd Qu.: 8.000
##  Max.   :41.00   Max.   :10.000   Max.   :5.000   Max.   :6.000   Max.   :10.000
##     MGODRK          MGODPR          MGODOV          MGODGE          MRELGE
##  Min.   :0.0000   Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.00
##  1st Qu.:0.0000   1st Qu.:4.000   1st Qu.:0.000   1st Qu.:2.000   1st Qu.:5.00
##  Median :0.0000   Median :5.000   Median :1.000   Median :3.000   Median :6.00
##  Mean   :0.7028   Mean   :4.616   Mean   :1.065   Mean   :3.275   Mean   :6.19
##  3rd Qu.:1.0000   3rd Qu.:6.000   3rd Qu.:2.000   3rd Qu.:4.000   3rd Qu.:7.00
##  Max.   :9.0000   Max.   :9.000   Max.   :5.000   Max.   :9.000   Max.   :9.00
##     MRELSA          MRELOV         MFALLEEN        MFGEKIND        MFWEKIND
##  Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.000
##  1st Qu.:0.000   1st Qu.:1.000   1st Qu.:0.000   1st Qu.:2.000   1st Qu.:3.000
##  Median :1.000   Median :2.000   Median :2.000   Median :3.000   Median :4.000
##  Mean   :0.881   Mean   :2.289   Mean   :1.894   Mean   :3.234   Mean   :4.298
##  3rd Qu.:1.000   3rd Qu.:3.000   3rd Qu.:3.000   3rd Qu.:4.000   3rd Qu.:6.000
##  Max.   :7.000   Max.   :9.000   Max.   :9.000   Max.   :9.000   Max.   :9.000
##     MOPLHOOG        MOPLMIDD        MOPLLAAG        MBERHOOG        MBERZELF
##  Min.   :0.000   Min.   :0.00   Min.   :0.000   Min.   :0.000   Min.   :0.0000
##  1st Qu.:0.000   1st Qu.:2.00   1st Qu.:3.000   1st Qu.:0.000   1st Qu.:0.0000
##  Median :1.000   Median :3.00   Median :5.000   Median :2.000   Median :0.0000
##  Mean   :1.476   Mean   :3.35   Mean   :4.556   Mean   :1.916   Mean   :0.3992
##  3rd Qu.:2.000   3rd Qu.:4.00   3rd Qu.:6.000   3rd Qu.:3.000   3rd Qu.:1.0000
##  Max.   :9.000   Max.   :9.00   Max.   :9.000   Max.   :9.000   Max.   :5.0000
##     MBERBOER        MBERMIDD        MBERARBG        MBERARBO          MSKA
##  Min.   :0.0000   Min.   :0.000   Min.   :0.00   Min.   :0.000   Min.   :0.00
##  1st Qu.:0.0000   1st Qu.:2.000   1st Qu.:1.00   1st Qu.:1.000   1st Qu.:0.00
##  Median :0.0000   Median :3.000   Median :2.00   Median :2.000   Median :1.00
```

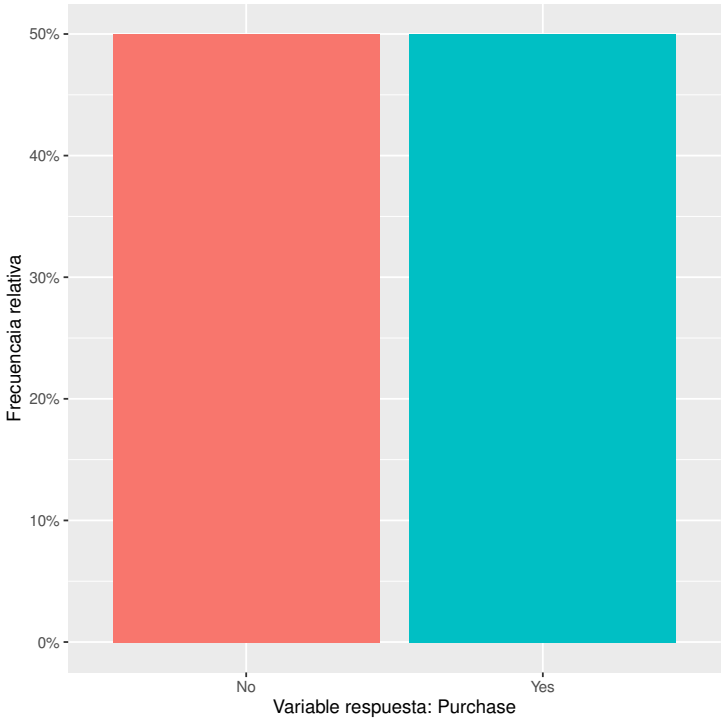
```
##  Mean   :0.5269   Mean   :2.893   Mean   :2.21   Mean   :2.285   Mean   :1.65
##  3rd Qu.:1.0000   3rd Qu.:4.000   3rd Qu.:3.00   3rd Qu.:3.000   3rd Qu.:2.00
##  Max.   :9.0000   Max.   :9.000   Max.   :9.00   Max.   :9.000   Max.   :9.00
##      MSKB1           MSKB2           MSKC           MSKD          MHHUUR
##  Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.000
##  1st Qu.:1.000   1st Qu.:1.000   1st Qu.:2.000   1st Qu.:0.000   1st Qu.:2.000
##  Median :2.000   Median :2.000   Median :4.000   Median :1.000   Median :4.000
##  Mean   :1.613   Mean   :2.185   Mean   :3.739   Mean   :1.062   Mean   :4.241
##  3rd Qu.:2.000   3rd Qu.:3.000   3rd Qu.:5.000   3rd Qu.:2.000   3rd Qu.:7.000
##  Max.   :9.000   Max.   :9.000   Max.   :9.000   Max.   :7.000   Max.   :9.000
##      MHKOOP          MAUT1           MAUT2          MAUT0         MZFONDS
##  Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.00   Min.   :0.000
##  1st Qu.:2.000   1st Qu.:5.000   1st Qu.:0.000   1st Qu.:1.00   1st Qu.:5.000
##  Median :5.000   Median :6.000   Median :1.000   Median :2.00   Median :7.000
##  Mean   :4.768   Mean   :6.037   Mean   :1.321   Mean   :1.96   Mean   :6.277
##  3rd Qu.:7.000   3rd Qu.:7.000   3rd Qu.:2.000   3rd Qu.:3.00   3rd Qu.:8.000
##  Max.   :9.000   Max.   :9.000   Max.   :7.000   Max.   :9.00   Max.   :9.000
##      MZPART        MINKM30         MINK3045        MINK4575        MINK7512
##  Min.   :0.00   Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.0000
##  1st Qu.:1.00   1st Qu.:1.000   1st Qu.:2.000   1st Qu.:1.000   1st Qu.:0.0000
##  Median :2.00   Median :2.000   Median :4.000   Median :3.000   Median :0.0000
##  Mean   :2.73   Mean   :2.582   Mean   :3.514   Mean   :2.744   Mean   :0.7973
##  3rd Qu.:4.00   3rd Qu.:4.000   3rd Qu.:5.000   3rd Qu.:4.000   3rd Qu.:1.0000
##  Max.   :9.00   Max.   :9.000   Max.   :9.000   Max.   :9.000   Max.   :9.0000
##     MINK123M         MINKGEM         MKOOPKLA        PWAPART          PWABEDR
##  Min.   :0.0000   Min.   :0.000   Min.   :1.000   Min.   :0.0000   Min.   :0.00000
##  1st Qu.:0.0000   1st Qu.:3.000   1st Qu.:3.000   1st Qu.:0.0000   1st Qu.:0.00000
##  Median :0.0000   Median :4.000   Median :4.000   Median :0.0000   Median :0.00000
##  Mean   :0.2049   Mean   :3.782   Mean   :4.253   Mean   :0.7677   Mean   :0.03736
##  3rd Qu.:0.0000   3rd Qu.:4.000   3rd Qu.:6.000   3rd Qu.:2.0000   3rd Qu.:0.00000
##  Max.   :9.0000   Max.   :9.000   Max.   :8.000   Max.   :3.0000   Max.   :6.00000
##     PWALAND          PPERSAUT        PBESAUT          PMOTSCO          PVRAAUT
##  Min.   :0.00000   Min.   :0.000   Min.   :0.00000   Min.   :0.0000   Min.   :0.000000
##  1st Qu.:0.00000   1st Qu.:0.000   1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.:0.000000
##  Median :0.00000   Median :5.000   Median :0.00000   Median :0.0000   Median :0.000000
##  Mean   :0.07193   Mean   :2.968   Mean   :0.05068   Mean   :0.1757   Mean   :0.005154
##  3rd Qu.:0.00000   3rd Qu.:6.000   3rd Qu.:0.00000   3rd Qu.:0.0000   3rd Qu.:0.000000
##  Max.   :4.00000   Max.   :8.000   Max.   :7.00000   Max.   :7.0000   Max.   :6.000000
##     PAANHANG         PTRACTOR          PWERKT           PBROM            PLEVEN
##  Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.00000   Median :0.00000   Median :0.00000   Median :0.0000   Median :0.0000
##  Mean   :0.02212   Mean   :0.08997   Mean   :0.01031   Mean   :0.2141   Mean   :0.1915
##  3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.0000   3rd Qu.:0.0000
##  Max.   :5.00000   Max.   :6.00000   Max.   :6.00000   Max.   :6.0000   Max.   :9.0000
##     PPERSONG         PGEZONG          PWAOREG          PBRAND          PZEILPL
##  Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.000   Min.   :0.000000
##  1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.000   1st Qu.:0.000000
##  Median :0.00000   Median :0.00000   Median :0.00000   Median :2.000   Median :0.000000
##  Mean   :0.01482   Mean   :0.01632   Mean   :0.02491   Mean   :1.824   Mean   :0.001074
##  3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:4.000   3rd Qu.:0.000000
##  Max.   :6.00000   Max.   :3.00000   Max.   :7.00000   Max.   :7.000   Max.   :3.000000
##     PPLEZIER          PFIETS          PINBOED          PBYSTAND         AWAPART
##  Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.0000
```

```
##  1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.0000
##  Median :0.00000   Median :0.00000   Median :0.00000   Median :0.00000   Median :0.0000
##  Mean   :0.02019   Mean   :0.02641   Mean   :0.01696   Mean   :0.05046   Mean   :0.4011
##  3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:1.0000
##  Max.   :6.00000   Max.   :1.00000   Max.   :6.00000   Max.   :5.00000   Max.   :2.0000
##     AWABEDR           AWALAND           APERSAUT          ABESAUT           AMOTSCO
##  Min.   :0.00000   Min.   :0.00000   Min.   :0.0000   Min.   :0.00000   Min.   :0.00000
##  1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:0.00000
##  Median :0.00000   Median :0.00000   Median :1.0000   Median :0.00000   Median :0.00000
##  Mean   :0.01374   Mean   :0.02061   Mean   :0.5632   Mean   :0.01138   Mean   :0.04166
##  3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:0.00000
##  Max.   :5.00000   Max.   :1.00000   Max.   :7.0000   Max.   :4.00000   Max.   :8.00000
##     AVRAAUT           AAANHANG          ATRACTOR          AWERKT
##  Min.   :0.000000   Min.   :0.00000   Min.   :0.0000   Min.   :0.000000
##  1st Qu.:0.000000   1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.:0.000000
##  Median :0.000000   Median :0.00000   Median :0.0000   Median :0.000000
##  Mean   :0.001074   Mean   :0.01331   Mean   :0.0335   Mean   :0.005154
##  3rd Qu.:0.000000   3rd Qu.:0.00000   3rd Qu.:0.0000   3rd Qu.:0.000000
##  Max.   :2.000000   Max.   :3.00000   Max.   :4.0000   Max.   :6.000000
##      ABROM             ALEVEN            APERSONG          AGEZONG
##  Min.   :0.00000   Min.   :0.00000   Min.   :0.000000   Min.   :0.000000
##  1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.000000   1st Qu.:0.000000
##  Median :0.00000   Median :0.00000   Median :0.000000   Median :0.000000
##  Mean   :0.06936   Mean   :0.07558   Mean   :0.005798   Mean   :0.006871
##  3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.000000   3rd Qu.:0.000000
##  Max.   :2.00000   Max.   :8.00000   Max.   :1.000000   Max.   :1.000000
##     AWAOREG            ABRAND           AZEILPL           APLEZIER
##  Min.   :0.000000   Min.   :0.0000   Min.   :0.0000000   Min.   :0.000000
##  1st Qu.:0.000000   1st Qu.:0.0000   1st Qu.:0.0000000   1st Qu.:0.000000
##  Median :0.000000   Median :1.0000   Median :0.0000000   Median :0.000000
##  Mean   :0.004939   Mean   :0.5714   Mean   :0.0006442   Mean   :0.006012
##  3rd Qu.:0.000000   3rd Qu.:1.0000   3rd Qu.:0.0000000   3rd Qu.:0.000000
##  Max.   :2.000000   Max.   :7.0000   Max.   :1.0000000   Max.   :2.000000
##      AFIETS            AINBOED           ABYSTAND       Purchase
##  Min.   :0.00000   Min.   :0.000000   Min.   :0.00000   No :4381
##  1st Qu.:0.00000   1st Qu.:0.000000   1st Qu.:0.00000   Yes: 276
##  Median :0.00000   Median :0.000000   Median :0.00000
##  Mean   :0.03307   Mean   :0.008374   Mean   :0.01503
##  3rd Qu.:0.00000   3rd Qu.:0.000000   3rd Qu.:0.00000
##  Max.   :3.00000   Max.   :2.000000   Max.   :2.00000
```

```
ggplot(data=car.train,aes(x=Purchase,fill=Purchase)) +
  geom_bar(aes(y=(..count..)/sum(..count..))) +
  scale_y_continuous(labels=scales::percent) +
  theme(legend.position="none") +
  ylab("Frecuencaia relativa") +
  xlab("Variable respuesta: Purchase")
```



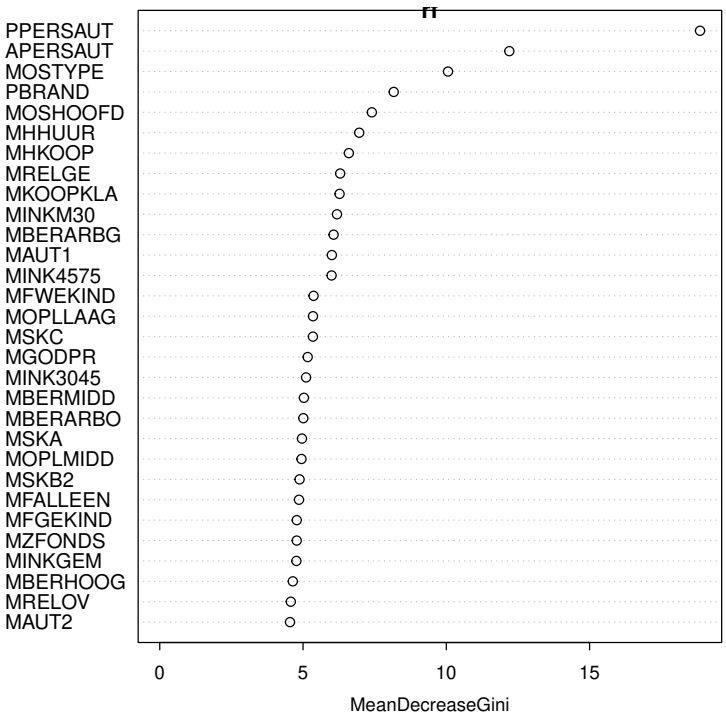

```
# Equilibramos las clases
train.yes=car.train[car.train$Purchase=="Yes",]
size1=dim(train.yes)[1]
train.no=car.train[car.train$Purchase=="No",]
dimension2=dim(train.no)[1]
indices.no=sample(1:dimension2,size=size1,replace=FALSE)
muestra.no=train.no[indices.no,]

car.train=rbind(car.train[car.train$Purchase=="Yes",],muestra.no)
ggplot(data=car.train,aes(x=Purchase,fill=Purchase)) +
  geom_bar(aes(y=(..count..)/sum(..count..))) +
  scale_y_continuous(labels=scales::percent) +
  theme(legend.position="none") +
  ylab("Frecuencaia relativa") +
  xlab("Variable respuesta: Purchase")
```



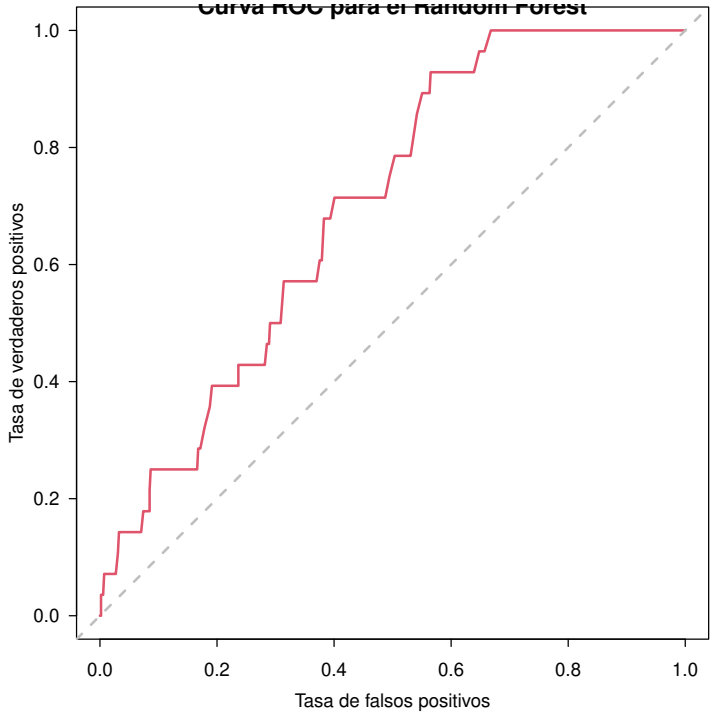

```
# Podemos comprobar como ahora la muestra de train está equilibrada

# Ajustamos un modelo de random forest
library(randomForest)
rf <-randomForest(Purchase~.,data=car.train, ntree=300)
print(rf)

##
## Call:
##  randomForest(formula = Purchase ~ ., data = car.train, ntree = 300)
##                Type of random forest: classification
##                      Number of trees: 300
## No. of variables tried at each split: 9
##
##         OOB estimate of  error rate: 34.24%
## Confusion matrix:
##      No Yes class.error
## No  183  93   0.3369565
## Yes  96 180   0.3478261

# Importancia de las variables
importance(rf)

##          MeanDecreaseGini
## MOSTYPE       10.05904963
## MAANTHUI       0.97148706
## MGEMOMV        2.99982901
## MGEMLEEF       2.73945263
## MOSHOOFD       7.40132377
## MGODRK         3.11540474
```

```
## MGODPR         5.16206064
## MGODOV         3.54183796
## MGODGE         4.50020156
## MRELGE         6.29591228
## MRELSA         3.42708164
## MRELOV         4.57335346
## MFALLEEN       4.85879716
## MFGEKIND       4.78080394
## MFWEKIND       5.36700709
## MOPLHOOG       4.48465020
## MOPLMIDD       4.94601049
## MOPLLAAG       5.34765518
## MBERHOOG       4.64327851
## MBERZELF       2.29936647
## MBERBOER       1.99476918
## MBERMIDD       5.03094781
## MBERARBG       6.06600522
## MBERARBO       5.00942241
## MSKA           4.96233239
## MSKB1          4.43889944
## MSKB2          4.87845474
## MSKC           5.34422043
## MSKD           3.49028301
## MHHUUR         6.96107866
## MHKOOP         6.59897367
## MAUT1          6.00579172
## MAUT2          4.54617054
## MAUT0          4.49778685
## MZFONDS        4.77888012
## MZPART         4.54065956
## MINKM30        6.18548906
## MINK3045       5.10631946
## MINK4575       5.99601714
## MINK7512       3.11406923
## MINK123M       2.40521110
## MINKGEM        4.76788814
## MKOOPKLA       6.27471357
## PWAPART        3.25969479
## PWABEDR        0.37903843
## PWALAND        0.38711742
## PPERSAUT      18.85519843
## PBESAUT        0.13948725
## PMOTSCO        0.63245819
## PVRAAUT        0.00000000
## PAANHANG       0.36411184
## PTRACTOR       0.34404056
## PWERKT         0.05130278
## PBROM          0.68501334
## PLEVEN         1.39712238
## PPERSONG       0.08259973
## PGEZONG        0.17193584
## PWAOREG        0.45610190
## PBRAND         8.16546475
## PZEILPL        0.04105962
```

```
## PPLEZIER       1.03885525
## PFIETS         0.59262307
## PINBOED        0.10883277
## PBYSTAND       0.64658945
## AWAPART        2.82835739
## AWABEDR        0.23497898
## AWALAND        0.41269745
## APERSAUT      12.19875224
## ABESAUT        0.13026212
## AMOTSCO        0.53350872
## AVRAAUT        0.00000000
## AAANHANG       0.49234946
## ATRACTOR       0.45963421
## AWERKT         0.04249799
## ABROM          0.74700286
## ALEVEN         1.65339999
## APERSONG       0.11538694
## AGEZONG        0.13805431
## AWAOREG        0.38868734
## ABRAND         2.40158370
## AZEILPL        0.04393799
## APLEZIER       1.15358862
## AFIETS         0.68422159
## AINBOED        0.11516453
## ABYSTAND       0.63127388

varImpPlot(rf)
```

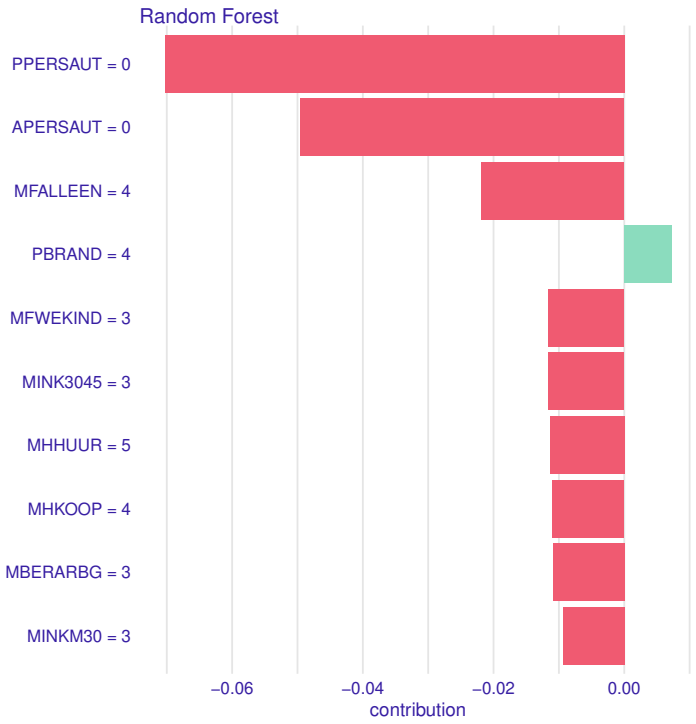


```r
# Curva ROC
pred1=predict(rf,car.test,type = "prob")
perf = prediction(pred1[,2], car.test$Purchase)
# True Positive y Negative Rate
pred3 = performance(perf, "tpr","fpr")
# ROC
plot(pred3,main="Curva ROC para el Random Forest",col=2,lwd=2,
     xlab="Tasa de falsos positivos",ylab="Tasa de verdaderos positivos")
abline(a=0,b=1,lwd=2,lty=2,col="gray")
```

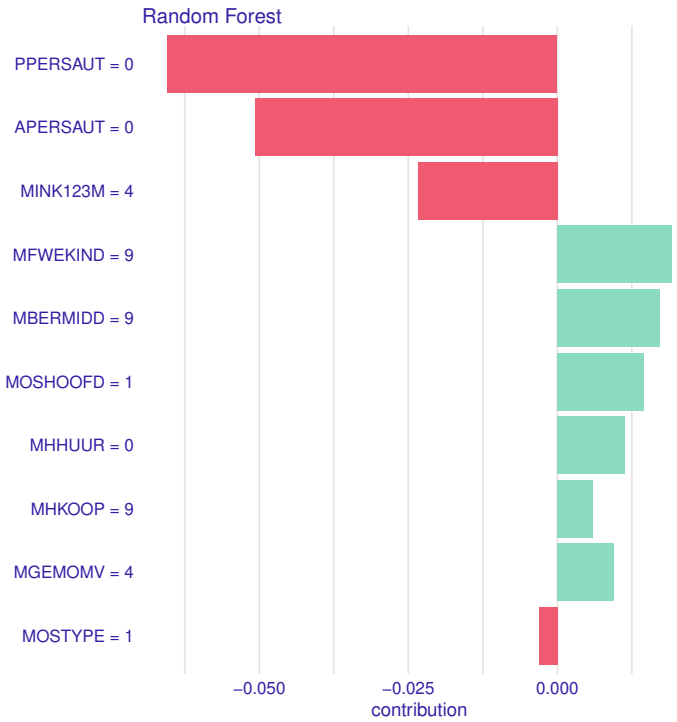



```r
explain_rf <- DALEX::explain(model = rf,
                             data = car.test,
                             y = car.test$Purchase,
                             label = "Random Forest")

## Preparation of a new explainer is initiated
##   -> model label       :  Random Forest
##   -> data              :  582  rows  86  cols
##   -> target variable   :  582  values
##   -> predict function  :  yhat.randomForest  will be used (  default  )
##   -> predicted values  :  No value for predict function target column. (  default  )
##   -> model_info        :  package randomForest , ver. 4.7.1.1 , task classification (  default  )
##   -> model_info        :  Model info detected classification task but 'y' is a factor .  (  WARNING
##   -> model_info        :  By deafult classification tasks supports only numercical 'y' parameter.
##   -> model_info        :  Consider changing to numerical vector with 0 and 1 values.
##   -> model_info        :  Otherwise I will not be able to calculate residuals or loss function.
##   -> predicted values  :  numerical, min =  0.01333333 , mean =  0.4199771 , max =  0.95
##   -> residual function :  difference between y and yhat (  default  )

## Warning in Ops.factor(y, predict_function(model, data)):  '-' not meaningful for factors
```

```
##   -> residuals         :  numerical, min =  NA , mean =  NA , max =  NA
##   A new explainer has been created!

obs1=car.test[1,]
obs2=car.test[2,]

predict(explain_rf, obs1)

## [1] 0.1933333

predict(explain_rf, obs2)

## [1] 0.32

shap_obs1 <- predict_parts(explainer = explain_rf,
                           new_observation = obs1,
                           type = "shap",
                           B = 25)

plot(shap_obs1, show_boxplots = FALSE)
```




```
shap_obs2 <- predict_parts(explainer = explain_rf,
                           new_observation = obs2,
                           type = "shap",
                           B = 25)
plot(shap_obs2, show_boxplots = FALSE)
```



Información de la sesión de R (incluyendo información sobre el sistema operativo, la versión de R y los paquetes usados):

```
sessionInfo()

## R version 4.3.1 (2023-06-16)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 20.04.6 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/atlas/libblas.so.3.10.3
## LAPACK: /usr/lib/x86_64-linux-gnu/atlas/liblapack.so.3.10.3;  LAPACK version 3.9.0
##
## locale:
##  [1] LC_CTYPE=es_ES.UTF-8       LC_NUMERIC=C               LC_TIME=es_ES.UTF-8
##  [4] LC_COLLATE=es_ES.UTF-8     LC_MONETARY=es_ES.UTF-8    LC_MESSAGES=es_ES.UTF-8
##  [7] LC_PAPER=es_ES.UTF-8       LC_NAME=C                  LC_ADDRESS=C
## [10] LC_TELEPHONE=C             LC_MEASUREMENT=es_ES.UTF-8 LC_IDENTIFICATION=C
##
## time zone: Europe/Madrid
## tzcode source: system (glibc)
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] DALEX_2.4.3          ROCR_1.0-11          randomForest_4.7-1.1 arulesViz_1.5-2
##  [5] arules_1.7-6         Matrix_1.6-1.1       liver_1.15           ggfortify_0.4.16
##  [9] factoextra_1.0.7     mlbench_2.1-3.1      readxl_1.4.3         caret_6.0-94
## [13] lattice_0.21-9       ggplot2_3.4.3        rpart.plot_3.1.1     rpart_4.1.19
## [17] caTools_1.18.2       dplyr_1.1.3          ISLR2_1.3-2
```

```
##
## loaded via a namespace (and not attached):
##  [1] bitops_1.0-7         pROC_1.18.4          gridExtra_2.3        rlang_1.1.1
##  [5] magrittr_2.0.3       e1071_1.7-13         compiler_4.3.1       vctrs_0.6.3
##  [9] reshape2_1.4.4       stringr_1.5.0        crayon_1.5.2         pkgconfig_2.0.3
## [13] fastmap_1.1.1        ellipsis_0.3.2       labeling_0.4.3       ggraph_2.1.0
## [17] utf8_1.2.3           rmarkdown_2.25       prodlim_2023.08.28   tzdb_0.4.0
## [21] tinytex_0.47         purrr_1.0.2          xfun_0.40            jsonlite_1.8.7
## [25] recipes_1.0.8        highr_0.10           tweenr_2.0.2         parallel_4.3.1
## [29] R6_2.5.1             stringi_1.7.12       parallelly_1.36.0    lubridate_1.9.3
## [33] cellranger_1.1.0     Rcpp_1.0.11          iterators_1.0.14     knitr_1.44
## [37] future.apply_1.11.0  readr_2.1.4          splines_4.3.1        nnet_7.3-19
## [41] igraph_1.5.1         timechange_0.2.0     tidyselect_1.2.0     rstudioapi_0.15.0
## [45] yaml_2.3.7           viridis_0.6.4        timeDate_4022.108    codetools_0.2-19
## [49] listenv_0.9.0        tibble_3.2.1         plyr_1.8.9           withr_2.5.1
## [53] evaluate_0.22        future_1.33.0        survival_3.5-7       proxy_0.4-27
## [57] polyclip_1.10-6      pillar_1.9.0         foreach_1.5.2        stats4_4.3.1
## [61] generics_0.1.3       hms_1.1.3            munsell_0.5.0        scales_1.2.1
## [65] globals_0.16.2       class_7.3-22         glue_1.6.2           tools_4.3.1
## [69] data.table_1.14.8    ModelMetrics_1.2.2.2 gower_1.0.1          visNetwork_2.1.2
## [73] graphlayouts_1.0.1   tidygraph_1.2.3      grid_4.3.1           tidyr_1.3.0
## [77] iBreakDown_2.0.1     ipred_0.9-14         colorspace_2.1-0     nlme_3.1-163
## [81] ggforce_0.4.1        cli_3.6.1            fansi_1.0.5          viridisLite_0.4.2
## [85] lava_1.7.2.1         gtable_0.3.4         digest_0.6.33        ggrepel_0.9.3
## [89] htmlwidgets_1.6.2    farver_2.1.1         htmltools_0.5.6.1    lifecycle_1.0.3
## [93] hardhat_1.3.0        MASS_7.3-60
```

```
Sys.time()
```

```
## [1] "2023-11-02 20:21:22 CET"
```